SIMULATION OF SOLAR ASSISTED HEAT PUMP SPACE HEATING SYSTEMS WITH SEASONAL THERMAL ENERGY STORAGES

A Ph. D. THESIS

in

Mechanical Engineering

University of Gaziantep

By

Recep YUMRUTAŞ

July 1998

Approval of the Graduate school of natural and Applied Sciences

Assoc. Prof. Dr. Ali Rıza TEKİN
Director V.

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Assoc. Prof. Dr. Melda Çarpınlıoğlu
Chairman of the Department

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Prof. Dr. Mazhar ÜNSAL
Major Supervisor

Examining Committee in Charge

Prof. Dr. Gürbüz ATAGÜNDÜZ (Chairman)
Prof. Dr. Ömer T. GÖKSEL
Prof. Dr. Mazhar ÜNSAL
Assoc. Prof. Dr. Melda ÇARPINLIOĞLU
Assist. Prof. Dr. M. Sait SÖYLEMEZ

ABSTRACT

SIMULATION OF SOLAR ASSISTED HEAT PUMP SPACE HEATING SYSTEMS WITH SEASONAL THERMAL ENERGY STORAGES

YUMRUTAŞ, Recep
Ph. D. in Mechanical Engineering
Supervisor : Prof. Dr. Mazhar ÜNSAL
July 1998, 295 pages

This study contains a simulation model for determining annual periodic performance of a solar aided space heating systems utilizing a heat pump and spherically and hemispherically shaped seasonal underground thermal energy storages.

The heating system investigated collects solar energy throughout the whole year and extracts thermal energy from the storage by a heat pump for space heating during the winter season. The monthly solar thermal energy collected is estimated using the $\bar{\phi}$ method. Two analytical solution procedures are given, one for prediction of the transient temperature field outside the spherical and another one outside the hemispherical energy storage. The solutions are obtained by applications of Complex Finite Fourier Transform, superposition, and Finite Hankel Transform techniques.

A numerical iterative procedure is used to determine the temporal variation of the seasonal thermal energy storage temperature and the thermal performance of the solar aided space heating systems under investigations. The present results are compared with the results from similar studies available in the literature.

Key Words: Solar energy, seasonal storage, heat pump, active solar heating.

ÖZET

MEVSİMLİK GÜNEŞ ENERJİSİ DEPOLAMALI VE ISI POMPALI KONUT ISITMA SİSTEMLERİNİN SİMÜLASYONU

YUMRUTAŞ, Recep
Doktora Tezi, Makina Mühendisliği Bölümü
Tez Yöneticisi : Prof. Dr. Mazhar ÜNSAL
Temmuz 1998, 295 sayfa

Bu çalışmada, yeraltındaki küresel veya yarıküresel tip mevsimlik güneş enerjisi depoları ve ısı pompası desteğinde konutların ısıtılmasında, ısıl sistemlerin yıllık periyodik performanslarının incelenmesi için bir model ve simülasyon verilmiştir.

İncelenen ısıtma sisteminde güneş enerjisinin yıl boyunca enerji deposuna aktarılması ve kış aylarında ise ısı pompası yardımı ile enerji deposundan çekilerek konutların ısıtılmasında kullanılmaktadır. Yıl boyunca her ay depolanan güneş enerjisi miktarı $\bar{\phi}$ yöntemi ile hesaplanmaktadır. Yer altında bulunan küresel veya yarıküresel şekildeki enerji deposu ve deponun çevresindeki jeolojik yapının geçici rejimdeki sıcaklığını hesaplamak için iki çözüm verilmiştir. Bu çözümler Kompleks Sonlu Fourier Dönüşümü, Süperpozisyon, ve Sonlu Hankel Dönüşümü yöntemlerinin uygulanmasıyla elde edilmiştir.

Yıllık periyodik depo su sıcaklığı ve güneş enerjisi destekli konut ısıtma sisteminin ısıl performansı bir mikrobilgisayar kullanarak hesaplanmış ve elde edilen sonuçlar grafiklerle gösterilmiştir. Bilgisayar simülasyonundan elde edilen sonuçlar literatürde bulunan önceki çalışmaların sonuçlarıyla karşılaştırılmıştır.

Anahtar Kelimeler: Güneş enerjisi, mevsimlik depolama, ısı pompası, güneşle aktif ısıtma.

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to Prof. Dr. Mazhar Ünsal for his encouragement, guidance, patience and many helpful suggestions throughout this study.

I thank to Instructor Murat Aksoy who ensured the Lahey F77L-EM/32 compiler and helped for installing the computer program. I also wish to thank to Assist. Prof. Dr. Nihat Yıldırım, Assist. Prof. Dr. Abdullah Akpolat and Assist. Prof. Dr. Ömer Eyercioğlu for their proofreading of this thesis and research assistant Necati Olgun and Tahsin Kılıç for his helping on the scan of some figures.

Special thanks are due to my dearest wife for her assistance in the preparation of the graphs and drawing, my dearest son M. Fatih Yumrutaş and my family for their moral support throughout the work.

TABLE OF CONTENTS

APPENDIX A

APPENDIX B

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURE

Symbol	Description
A	Parameter defined on page 51
A_c	Collector area
A_s	Surface area of storage
A_T	Top surface area for hemispherical storage
a	Radius of spherical storage
a_n	Constant defined on page 81
a_1	Parameter defined on page 53
B	Parameter defined on page 51
B_i	Biot number
$B_{i\theta}$	Parameter defined on page 68
b_{nm}	Constant defined on page 81
b_0	Incidence angle modifier
b_1	Parameter defined on page 53
C	Parameter defined on page 51
c	Specific heat of soil
c_{nm}	Constant defined on page 81
c_w	Specific heat of water
COP	Coefficient of performance of heat pump
del R	Distance from storage
f	Annual solar fraction
F_R	Collector overall heat removal factor
H	Daily total radiation on a horizontal surface
$\bar{H}$	Monthly average daily total radiation on a horizontal surface
H_d	Daily diffuse radiation on a horizontal surface
$\bar{H}_d$	Monthly average daily diffuse radiation on a horizontal surface

$\bar{H}_T$	Monthly average daily total radiation on a tilted surface
$H_{t,c}$	Critical radiation level
$\bar{H}_0$	Monthly average daily extraterrestrial radiation on a horizontal surface
$H^{(1)}_{\mu_n}$	Hankel functions of first kind of real order μ_n
$H^{(2)}_{\mu_n}$	Hankel functions of second kind of real order μ_n
$H^{(1)}_{\mu_{n+1}}$	Hankel functions of first kind of real order μ_{n+1}
$H^{(2)}_{\mu_{n+1}}$	Hankel functions of second kind of real order μ_{n+1}
i	Complex argument
J_0	Bessel function of first kind, of zero order
J_1	Bessel function of first kind, of first order
J_{μ_n}	Bessel function of first kind of real order μ_n
$J_{\mu_{n+1}}$	Bessel function of first kind of real order μ_{n+1}
K	Kernel of eigenvalue problem
K_T	Daily clearness index
$\bar{K}_T$	Monthly average clearness index
k	Thermal conductivity of earth
k_m	Thermal conductivity of insulating material
n_d	number for average day of a month
p	$(1-\cos\theta)\ \rho_w c_w/(3\rho c)$
p_s	$\rho_w c_w/(3\rho c)$
Q	Net energy charge rate to the store
Q_h	Rate of heat loss from house
Q_d	Design house heat load
Q_u	Instantaneous useful heat gain of solar system
q	$Q/(4\pi a k T_\infty)$, dimensionless net energy charge rate to the store
q_h	$Q_h/(4\pi a k T_\infty)$, dimensionless rate of heat loss from the house
q_{hp}	Dimensionless instantaneous energy extracted by heat pump

q_{hy}	Annual dimensionless house heat load
q_i	Value of q during i^{th} month
q_k	Complex Fourier coefficient of dimensionless heat input rate to store
q_{ls}	Annual dimensionless energy loss rate
q_{sy}	Annual dimensionless solar heat gain rate
q_u	$Q_u/(4\pi akT_\infty)$, dimensionless solar heat gain rate
q_y	Annual dimensionless net heat addition rate to the thermal store
$\bar{R}$	Ratio of monthly average radiation on tilted surface to that on a horizontal surface
R_b	Ratio of daily beam radiation on a tilted surface to that on a horizontal surface
$\bar{R}_b$	Ratio of monthly average daily beam radiation on a tilted surface to that on a horizontal surface
$R_{b,n}$	Ratio of beam radiation on a tilted surface to that on a horizontal surface at noon
R_H	Annual house heat loss
R_L	Annual energy loss
R_n	Ratio of radiation on a tilted surface to that on a horizontal surface at noon
R_s	Annual solar energy
R_w	Annual heat pump work
R_∞	Far field radial distance
r	Radial coordinate
$r_{d,n}$	Ratio of hourly diffuse radiation at noon to daily diffuse radiation
$r_{t,n}$	Ratio of hourly diffuse radiation at noon to daily total radiation
r_j	(j-6)/12
s	Dimensionless parameter defined on page 68
t	Time
T	Temperature of earth surrounding the thermal storage

Teta	Polar angle defined for figures
$\overline{T}_a$	Monthly average ambient air temperature
$\overline{T}_{ci}$	Monthly average collector fluid inlet temperature
T_d	Design outside air temperature
T_h	Temperature of fluid in load side heat exchanger.
T_i	Inside air temperature
T_w	Temperature of water in the thermal storage
T_∞	Far field deep earth temperature
u	$(UA)_h/(UA)_{he}$
$(UA)_h$	UA-value of the house
$(UA)_{he}$	UA-value of the load side heat exchanger
U_C	Collector overall heat loss coefficient
U_T	Top loss coefficient from hemispherical store
V_{ws}	Volume of spherical store
V_w	Volume of hemispherical store
W	Heat pump power
w	$W/(UA)_h T_\infty$ dimensionless heat pump power
w_s	Sunset hour angle for a horizontal surface
w'_s	Sunset hour angle for a tilted surface
$\overline{X}_C$	Monthly average dimensionless critical radiation ratio
x	r/R, dimensionless radial distance
x_∞	R_∞/R, dimensionless far field radial distance
Y	α(one year)$/a^2$, dimensionless time for one year
Y_{μ_n}	Bessel function of second kind of real order μ_n
$Y_{\mu_{n+1}}$	Bessel function of second kind of real order μ_{n+1}

Greek symbols

α	Thermal diffusivity of earth
α_n	Eigenvalues defined on page 81
α_R	Relaxation constant
γ	$4\pi ak/(UA)_h$
β	Collector slope
β_{hp}	Parameter defined on page 42
β_n	Parameter defined on page 72
θ	Polar angle
θ_b	Incident angle for beam radiation
θ_d	Incident angle for diffuse radiation
θ_i	Incident angle of solar radiation
θ_r	Incident angle for reflected radiation
θ_r	Incident angle for reflected radiation
θ_0	Angle defined on page 66
θ_1	Angle defined on page 66
ρ	Density of earth
ρ_w	Density of water
ρ_g	Ground reflectance
τ	Dimensionless time
τ_p	α(one year)/R^2, dimensionless time for one year
$(\tau\alpha)$	Transmittance-absorbtance product
$(\tau\alpha)_b$	Transmittance-absorbtance product for beam radiation
$(\tau\alpha)_d$	Transmittance-absorbtance product for diffuse radiation
$(\tau\alpha)_r$	Transmittance-absorbtance product for reflected radiation
$(\overline{\tau\alpha})_b$	Monthly average transmittance-absorptance product for beam radiation
$(\overline{\tau\alpha})_d$	Monthly average transmittance-absorptance product for diffuse radiation

$(\overline{\tau\alpha})_r$	Monthly average transmittance-absorptance product for reflected radiation
$(\overline{\tau\alpha})$	Monthly average transmittance-absorptance product
$\bar{\phi}$	Monthly average daily collector utilizability
ϕ_a	$(T_a-T_\infty)/T_\infty$, dimensionless ambient air temperature
ϕ_i	$(T_i-T_\infty)/T_\infty$, dimensionless design inside air temperature
ϕ_L	Latitude angle
ϕ_w	$(T_w-T_\infty)/T_\infty$, dimensionless water temperature in the storage
ϕ_{wn}	Complex Fourier coefficient of dimensionless water temperature in the storage for the one dimensional problem
ϕ_{wk}	Complex Fourier coefficient of dimensionless water temperature in the storage for the two dimensional problem
Φ	Eigenfunction defined on page 80
Φ_n	Normalized eigenfunctions defined on page 80
Φ_m	Normalized eigenfunctions defined on page 81
Φ_0	Normalized eigenfunctions for n=0 defined on page 80
ψ	Dimensionless temperature in earth
ψ_n	Complex Fourier coefficient of dimensionless earth temperature for one dimensional problem
ψ_k	Complex Fourier coefficient of dimensionless earth temperature for two dimensional problem
ψ_{1k}	Complex Fourier coefficient of dimensionless temperature defined on page 71
ψ_{1kn}	Complex Fourier coefficient of dimensionless temperature defined on page 71
ψ_{2k}	Complex Fourier coefficient of dimensionless temperature defined on page 71
η_c	Collector efficiency
η_1	Parameter defined on page 41
η_2	Parameter defined on page 41

$\eta_{3,j}$	Parameter defined on page 44
$\eta_{4,j}$	Parameter defined on page 44
Θ_1	Parameter defined on page 77
Θ_2	Parameter defined on page 77
φ	Parameter defined on page 47
ω_k	Complex argument defined on page 74
μ_n	Order of Hankel functions defined on page 74
λ_n	Eigenvalues defined on page 81
ν_k	$2\pi k/\tau_p$
ε	Parameter defined on page 79

CHAPTER I

INTRODUCTION

Energy has always been the most important factor in the progress of the humanity. Today, dependence on energy is so extensive that most functions of a society go to a complete stop in the event of a power failure or fuel shortage. In recent years, energy conservation has gained importance because of increasing energy consumption, rapid rising of fuel costs following the 1970 World oil crises, and concerns related to environmental pollution problems. Energy and economy are closely linked: increased production and wealth means greater consumption of energy. It is also known that increases in living standards of a society can only be achieved by increasing the energy consumption per capita. Furthermore, economic growth rate is roughly proportional to the energy utilisation rate. Turkish economy, like that of other developing countries, has become highly dependent upon fossil fuels such as coal and petroleum. In the World, large amount of all energy consumed is presently processed in thermal power plants by combustion of fossil fuels or by fission. The combustion of the fossil fuels causes pollution, and increase in noise levels. Energy sources are depleted consumption rates with growing world's population and increasing energy by developing industry.

It is necessary to search for new renewable energy sources to eliminate all negative effect of fossil fuel consumption on human and all other environmental life. Research on alternative renewable energy sources having no pollution by products has gained momentum in recent decades. Solar energy is the major alternative energy source for low temperature heating applications and it is a potentially feasible solution to the pollution, noise and economical energy related problems. The most common applications of low temperature solar energy using flat plate solar collectors are domestic water heating, residential space heating, solar air conditioning and solar industrial process heating. Solar domestic water heating systems are widely used in Turkey utilizing the high amount of solar radiation especially in southern regions. Solar aided space heating systems however, are not economically feasible when used for direct space heating. Solar energy is an intermittent energy source because of seasonal variations. Availability of excess solar energy in summer months and lack of energy required for heating of residences during winter months is a situation of mismatch between building energy demand and solar energy supply. Seasonal storage of solar energy during summer months and use of the stored energy for space heating applications during winter months is one way of compensating for the mismatch between energy consumption and production. In this way, it may be possible to increase solar aided heating applications.

The storage of solar energy is necessary to realize a substantial solar contribution in solar aided space heating systems. The solar thermal energy or energy end products obtained from solar processes can be stored as electrical, chemical, or mechanical energy. Three fundamental types of thermal energy storage are sensible, latent, and thermochemical. Some of these methods need further research and development to make them technically and economically feasible. One of the highly potential type of the energy storage is the storage of solar thermal energy. The amount of the sensible energy stored is a function of the temperature change, the mass of the storage medium, and its specific heat.

The choice of storage media depends to a large extent on the nature of the solar thermal process, availability of solar radiation and the economic assessment of solar versus auxiliary energy supplies. The space and other requirements for a given storage capacity are determined mostly by the physical and chemical properties of the storage medium employed. Sensible heat systems employ water, rocks, earth or ceramic bricks as the thermal storage material and water, air, or possibly oil as the heat transfer fluid. In water heating applications, water is the most commonly used sensible heat storage medium in solar energy applications because it possesses numerous desirable qualities, such as low cost, readily availability and high specific heat (4.18 kj/kg C). The relatively low heat capacity of rock and ceramics(in the order of 0.84 kj/kgC) is somewhat offset by the large temperature changes possible with these materials and their relatively high density. It should be noted that water has three times the heat capacity of rock on a volume basis, meaning that rock requires three times more volume than water to store the same amount of sensible heat. Variations of these storage processes are being researched and developed to improve thermal energy storage implementation, focusing on applications in building heating and cooling, industrial energy systems, as well as utility and space power systems. Thermal energy can contribute significantly to meeting society's needs for more efficient, environmentally benign energy use in these and other sectors.

The seasonal storage of solar thermal energy in underground vessels for space heating purposes with a heat pump has been the subject of many previous investigations and has also found practical applications in the past. A ground coupled heat pump heating system uses a heat exchanger buried in the ground. Because the seasonal variation of temperature in the ground is small relative to the variation in air temperature, the ground is normally at a more favourable source temperature than the outside air. In order for ground coupled heat pumps to compete successfully with conventional systems, the thermal conductivity of the soil must be high enough to transfer the required amount of heat as effectively as the fan-coils in air-to-air heat pumps. For seasonal thermal energy storage, several different approaches have been suggested in literature. Boreholes in rock, rock caverns, underground steel tanks, aquifers, open pits on earth, abandoned mines, abandoned hydropower tunnels, horizontal and vertical pipes embedded in earth, underground concrete tanks and excavated bedrock are examples of physical space for seasonal thermal energy storage.

Many research work and applications on the ground coupled heat pump systems has been accomplished in the last 30 years. Studies based on analytical modelling and simulation of the seasonal thermal storage system, however, are limited in the literature. Existing reviews on this subject are surveyed and explained in chapter II. In this study, two analytical solution models of the transient temperature field problem outside buried spherically and hemispherically shaped underground thermal energy storages are presented. The temperature field outside the storages is analyzed as a one dimensional problem for the spherical storage and a two dimensional problem for the hemispherical storage. A computer simulation program is developed for predicting annual periodic performance of a solar assisted heat pump space heating system with spherical and hemispherical type underground thermal energy storage.

This thesis consists of six chapters which are organised in such a way that; in the second chapter, comprehensive literature survey of research on the subject is summarized. Historical background on the concept of using the ground as a heat source for a heat pump, basic ground coil design configurations, theoretical and experimental studies on the seasonal storage of solar energy in earth, and results from these studies, ground coupled heat pump heating systems, energy extraction rates from underground heat exchangers are reviewed.

In the third chapter, an analytical modeling procedure is presented for predicting the annual periodic performance of solar aided space heating system utilising a heat pump and a spherical seasonal underground thermal energy storage. The problem formulation consisting of a differential equation, boundary conditions and energy balance for the spherical storage is put into dimensionless form by introducing dimensionless variables. Complex finite Fourier transform is applied to the dimensionless form of the problem, and the problem is solved assuming monthly heat input rates to the storage. Estimation of solar energy collection, the monthly solar heat gains, by collectors is obtained using the $\bar{\phi}$ method.

An analytical solution of the annually periodic transient heat transfer problem outside the hemispherical thermal energy storage is presented in chapter 4. The problem consists of a differential equation, boundary conditions, and energy equation for the storage which is formulated using the spherical coordinate system. Problem formulation is transferred into dimensionless form by introducing dimensionless variables. Complex finite Hankel and finite Fourier transforms are applied to the dimensionless formulation of the problem. Dimensionless formulation of the problem is transformed and solved as a function of complex Fourier coefficient of the dimensionless monthly energy input rates to the storage. Finally, estimation of the complex Fourier coefficients and the solution of the associated eigenvalue problem in the θ-direction is given in this chapter.

Chapter 5 consists of three sections. In the first section, results obtained from execution of the computer simulation program for solar aided ground coupled heat pump system with underground seasonal one dimensional spherical energy storage are depicted in graphical forms and the results are discussed. In the second section, results obtained from execution of the computer simulation program for the solar aided heating system with underground seasonal two dimensional hemispherical energy storage are given in figures and the results are discussed. The third section contains comparisons of results from this investigation and those from similar studies available in the literature.

Conclusions of this investigation and recommendations for further study are presented in chapter six.

CHAPTER II

LITERATURE SURVEY

2.1. INTRODUCTION

Storage of solar thermal energy plays an important role in heating systems if an efficient and cost-effective solar energy utilization can be achieved. One of the most promising methods of storage uses undisturbed ground as the storage medium, but to make ground heat storage feasible, construction costs must be lowered, energy transfer mechanisms into and out of the storage system must be improved, and heat losses to the surrounding ground and ambient air must be minimized. Thus, there are many problems to overcome the uncertainties in designing and installing ground-coupled heat pump systems.

One of the most promising possibilities is to use natural ground as a low grade heat storage and to employ a heat pump to upgrade the energy level and improve ground coupling energy exchange. This chapter deals with literature review of developments in thermal energy storage systems and use of the stored energy in solar aided heating systems with a heat pump, focusing on problems related to the designing, installing and performance of ground coupled heat pump heating systems.

2. 2. HISTORICAL BACKGROUND

Seasonal storage of thermal energy obtained from solar collectors is not a new idea. The idea of using undisturbed ground as a heat source for heat pump applications was introduced in a Swiss patent by Zoelly in 1912

(Wirth[1]. After the Second World War, particularly following the energy crisis in the early 1970s this concept was reconsidered for commercial use. Kemler[2] and Crandall[3] introduced the Zoelly 's plan. Ingersol and Plass[4] developed the classic theory for ground pipe heat conduction. They adapted Kelvin 's heat source theory for flow of heat from soil to buried pipes. Handly[5] correlated operating data of existing heat pump test installations. Hooper[6] reported the first research installation in Canada.

The end of the 1940s and the first half of the 1950s was marked by the installation of a large number of experimental ground-coupled heat pumps. Coogan[7], Smith[8], Penrod et al. [9], Kidder and Neher[10] and Harlow and Klapper[11] published experimental results in the United States at this period. These researches provided some actual information on the heat extraction rates in practical applications. For example, Penrod[12] obtained a heat flux of up to 50 W/m for 25 mm diameter copper pipes buried horizontally at a depth of 1.5 m. Vestal and Fluker[13] achieved similar heat extraction rate for copper pipes 13 and 25 mm in diameter buried at a depth of 1.6 m in various soils in Texas. They extracted up to 100 W/m with larger (50 mm) diameter copper pipes during 7 days. Similar results were published by Smith[14], and Freund and Whitlow[15]. Canadian researcher Hooper[6] reported seasonal average extraction rate about 21 W/m from 1.8 cm horizontally laid copper coils.

In Europe during the early 1950s, the first ground-coupled heat pump installation in Great Britain was designed and built by Griffith[16] who determined rates of heat extraction using horizontal copper pipes buried in clay at depths of 1.25, 1.8, and 2.5 cm in London. He reported heat extraction rates of 30-60 W/m for steady state conditions and 2 hour transient heat flows respectively. Another U. K. researcher Sumner[17] installed ground coupled systems on residential buildings. At about the same time German researcher, von Cube installed, in his own home, a horizontal steel pipe system that is still in operation. Von Cube and Stiemle[18] recorded an annual average coefficient of performance of about 3. The von

Cube installation uses direct expansion of R-22 in horizontal pipes placed 0.5 to 0.8 m below the ground level. The experience accumulated in this installation has been of great use in this type of systems.

In the late 1970s, Northern Europeans began taking a serious look at seasonal storage systems. They were accustomed to viewing building heating systems within the framework of a district heating utility. The early northern European experience evolved into an international research collaboration on Central Solar Heating Plant with Seasonal Storage (CSHPSS) within the International Energy Agency's (IEA) Solar Heating and Cooling Program started in 1970. Since then, more than 30 projects have been constructed and are now operating. Locations and characteristics of these projects was published by Breger et. al.[19]. These are listed in Table 1. In Sweden, the construction and monitoring of 13 projects have brought CSHPSS close to commercial readiness (Dalenback[20]). A new Swedish project is being planned to serve a small town (2000 homes). Studies also indicate that at least 100 to 150 similar plants could be built within the next few decades. The next period of intense activity on ground coil heat pumps started in Europe in 1974, following the first organization of Petroleum Exporting Countries Oil Price rise, and continues to the present. Ball et al.[21] researched ground coil system installations. In Europe, over 30 research and development projects concerning the development of design methods, installation techniques, and operating experience have been identified. In addition, over 6000 horizontal ground coil systems have been installed in Sweden, and with an unknown but significant number of similar installation in Germany.

AEIC-EEI Heat Pump Committee (1953) published review of earth heat pump installations in the U.S., It was proposed in the U.S. in the 1960s and research projects were conducted in the 1970s. These early studies were applied to single residences or small groups of houses. They were found to have low efficiency because of the small seasonal thermal storages. It showed interesting results for heat transfer on two different surfaces which are horizontal tubes and vertical tanks. For horizontal tubes, the ranges of heat transfer were 1.2-8.9 W/m°C and 1.1-2.3 W/m°C for heating and cooling cycles, respectively. For vertical surfaces, the ranges were 6.1-38.2 W/m^2°C and 8.3 W/ m^2°C again for heating and cooling cycles.

Table 2.1. Summary data on operating Central Solar Heating Plants with Seasonal Storage (CSHPSS) around the world, Breger et al.[19]

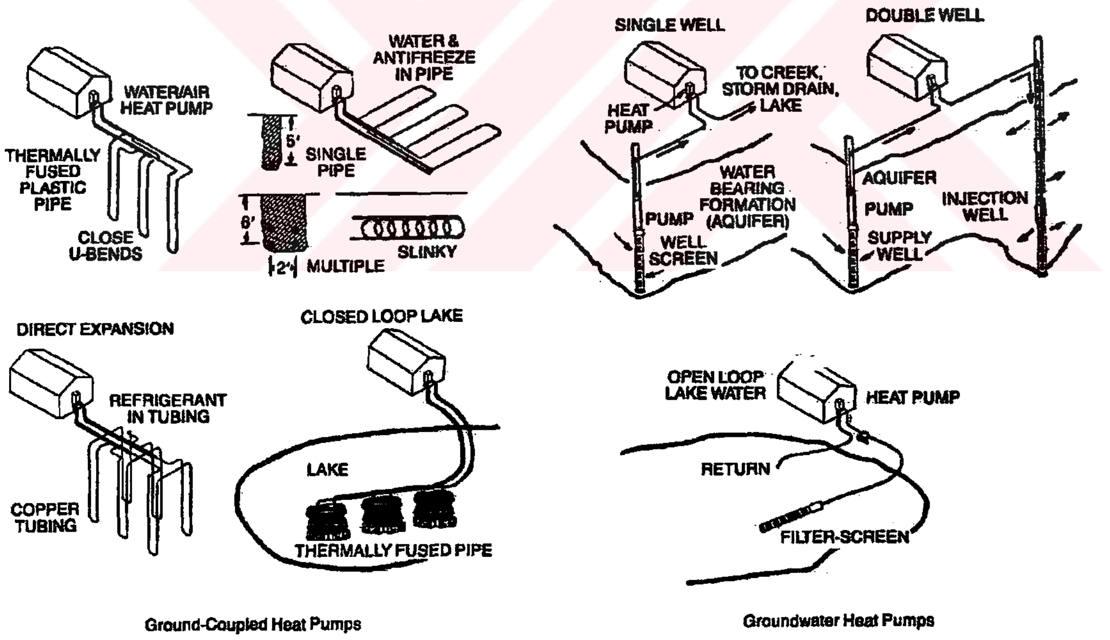
			Storage		Collectors		Load		
							Annual	Delv.	
		Year		Volume		Area	Load	Temp.	Heat
Country	Location	Built	Type	(m3)	Type	(m2)	(Mwh)	(C)	Pump
Austria	Kranebitten	1982	Tubes/earth	7200	Unglazed	416	1370		Yes
Canada	Scarborough	1985	Aquifer	250000	Evacuated	1300	1800		Yes
Denmark	Lyngby	1983	Pit	500	Non solar				
Finland	Kerava	1983	Pit & duct/rock	11000	Flat plate	1100	550	60	Yes
France	Toulous-Blangna	1980	Tank/concrete	200	Flat plate	110			
France	Aulnay-Sous-Boi	1983	Aquifer	85000	Unglazed	1275	2473	50	Yes
France	Saint Quentin	1986	Aquifer	1500	Unglazed	300			
France	Cormontreuil	1986	Duct/chalk	15000	Flat plate	110			
Germany	Stuttgart	1985	Aquifer/pit	1050	Unglazed	211	175	60	Yes
Italy	Treviglio	1982	Duct/earth	43000	Flat plate	2000	1235	45	Yes
Italy	Ispra	1981	Duct/earth	300	Flat plate	180			Yes
Japan	Yonesewa	1982	Aquifer		Unglazed	240			Yes
Nethar.	Groningen	1984	Duct/clay	23000	Evacuated	2400	1200	43	
Nethar.	Bunnik	1985	Aquifer	35000	Flat plate	450	660	50	Yes
Sweden	Studsvik	1979	Earth pit	640	CPC	120	23	30	
Sweden	Ingeslad	1979	Tank	5000	Parabolic	1300	940	80	
Sweden	Sunclay	1980	Duct/clay	87000	Unglazed	1500	1650	55	Yes
Sweden	Lambohov	1980	Rock pit	10000	Flat plate	2900	940	55	Yes
Sweden	Gullspang	1982	Tunnel	10000	Lake				Yes
Sweden	Lyckebo	1983	Rock cavern	100000	Flat plate	4300	7000	70	Yes
Sweden	Lulea	1983	Duct/rock	100000	Non Solar		2000		Yes
Sweden	Vallentuna		Lake sedim.	1200000	Lake				Yes
Sweden	Kopparberg		Aband. mine	240000	Lake		4200		Yes
Sweden	Klippan		Aquifer/sand	800000	River		16000		Yes
Sweden	Harryda		Tubes/earth	18000	Unglazed	550			Yes
Sweden	Suncourt		Duct/rock	30000	Atrium		320		Yes
Sweden	Falun		Aquifer/gravel	700000	Lake				Yes
Switzer.	Cort. Neuchatel	1981	Duct/earth	4500	Flat plate	320	190	60	Yes
Switzer.	Lagigny-Vaud	1981	Duct/sand	3500	Flat plate	50			
Switzer.	Vaulruz	1982	Duct/ earth	3500	Flat plate	520	341	50	Yes
Switzer.	Meyrin-Generva	1986	Duct/ earth	20000	Flat plate	550	1000		Yes

As we see from the above summarized research work in the world, there has been many ground-coil applications which were installed in ground coil heat pump systems. The ground-coil configurations used in heating systems will be explained in next section.

2. 2.1. Ground Coil Configurations

The development of ground coupled heat pump systems initially occurred in rural, and residential areas where heating requirements were the primary considerations. Recent improvements in heat pump systems and installation procedures have expanded the market to urban and commercial areas in many parts of the world. Closed loop ground coupled heat pumps have been receiving increasing attention in commercial applications.

There are a variety of names for ground coupled heat pump systems. These include ground source heat pumps, earth coupled heat pumps, earth energy systems, ground source systems, geothermal heat pumps, closed loop heat pumps and solar energy heat pumps. Different ground coupled heat pump systems are shown in Figure 2.1 as given by Kavanaugh[22].



Ground-Coupled Heat Pumps

Groundwater Heat Pumps

Figure 2.1. Ground source heat pump types, Kavanauh [22]

The most common method of ground coupling is to bury plastic pipes and this is classified into two categories. These are vertical and horizontal pipe systems. A water or antifreeze solution is circulated through the tubing and heat is released to or absorbed from the ground. A wide variety of possibilities exist within each category for specific geometry, installation method, material construction, and choice of circulating fluid through the ground coil. Horizontal systems are installed relatively close to the surface, Vertical systems on the other hand generally extend to a depth of 10 m or more, where the ground conditions are not affected by changes at the surface.

2. 2. 1. 1. Horizontal Coils

The single layer horizontal ground coil is the most common of all ground-coil systems in use today. The ground coil typically consist of circular plastic or metal tubes of external diameters between 20 mm and 50 mm laid at depths from 0.5 m to 2.5 m. Tube spacing vary from 0.6 m to 2.5 m.

Most modern systems use the serpentine arrangement. This arrangement is especially attractive when flexible polyethylene or polybutylene tubing is used with low-cost trenching techniques. Flexible tubing allows for minimum number of joints and, leakage problems, as well as the use of low cost trenching techniques.

A recent advance in horizontal coil design involves installing a double layer, one at about 1.2 m depth and the other at about 1.9 m depth. This technique is being employed in Belgium (Geeraert et al.[23]) in Okalahoma.

2. 2. 1. 2. Vertical Coils

Vertical ground coils may be either shallow or deep. Shallow coils are more common in Europe than in the U. S. and are placed at a depth of 8 to 10 m deep. These use flexible plastic coaxial tubes and are arranged in circular, hexagonal, or rectangular matrices. Circular arrangements are preferred because the cylindrical storage area formed has a minimal surface area and consequently, minimal heat loss. These benefits are particularly important for systems charged with heat beyond the normal average ground temperature.

Deep vertical coils are being installed in Germany, Sweden, and the U. S. The German approach involves reaching ground water so that the heat extraction rate per unit of tube length can be considerably increased. The coaxial tubes are placed between 50 and 100 m deep, depending on the depth of ground water, type of soil etc. The Swedish installation, near Goteburg, uses flexible plastic U-tubes placed 35 m deep spaced 2 m between tubes and low temperature solar collectors for regeneration. This arrangement permits a high storage density, a high storage temperature, and consequently, a high COP for the heating system.

In the U. S., deep vertical systems are installed to a depth of about 73 m in a drilled hole. A 10 to 12.5 cm PVC pipe is flooded with water and heat is exchanged with the circulating fluid by direct contact or through the walls of an immersed U-tube having one insulated leg to prevent short circulating.

2. 3. COMPREHENSIVE SUMMARY OF THE LITERATURE

In this section, comprehensive summary of the literature survey is presented about research on storage of solar energy in earth for various storage types, ground-coupled heat pump systems, solar assisted space heating systems and extraction of heat from the earth.

Ground-coupled heat pump systems are more common compared to solar aided heating systems. In recent years, the sub storing of solar energy in the summer season and using the stored energy in the winter season has been considered by many investigations. Taşdemiroğlu[24] noted a scheme for the solar energy storage methods which is shown in Figure 2.2.

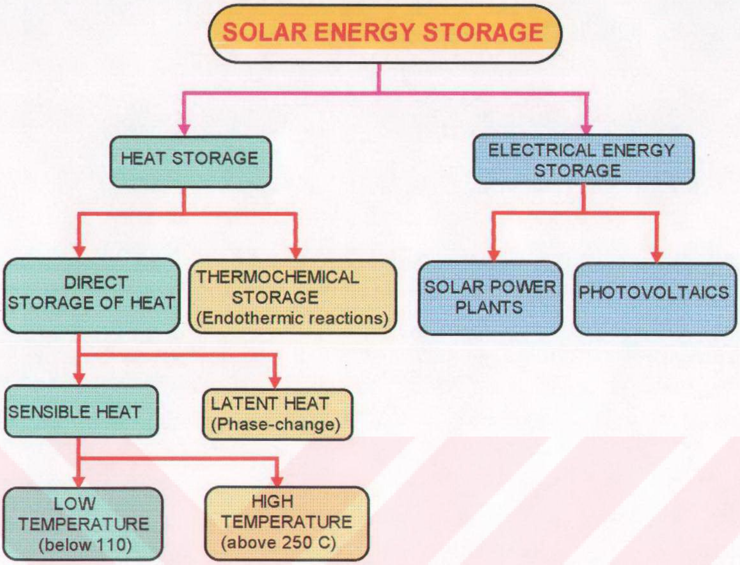


Figure 2.2. Solar energy storage types, Taşdemiroğlu[24]

Lund[25] discussed optimization of a district solar heating system with an electric-driven heat pump and seasonal heat storage. The optimization process comprises thermal, economic and system control analysis. He derived thermal and economic optima for collector area and storage volume. He also investigated effects of different collector types and building loads. He found that a system's cost effectiveness is only slightly affected as storage volume is increased beyond the optimum storage size and found that large variation occurs in the optimum size for different system configurations. He estimated 8.9 c/kWh minimum cost of heat supplied in an optimal 500-unit community with 90 % solar fraction.

Reid et al.[26] measured overall system thermal performance and electric peak demand for two configurations of series solar assisted heat pumps in TECH House I of the University of Tennessee Campus in Knoxville, Tennessee. The heating system consists of 62.7 m^2 of liquid type solar collectors, two different thermal storage units, a heat pump and a space to be heated. The first thermal storage was a 4.2 m^3 insulated concrete tank buried just below the surface of the ground in the crawl space. This tank was used during the 1979-1980 heating season. During the 1980-1981 heating season, a second storage unit was employed. It consisted of a 7.6 m^3 uninsulated steel tank buried, horizontally, 2.1 m deep in the ground. Water used in either tank is treated with corrosion inhibitors and microbicide.

They found that system performance factor was 2.42 and the system's maximum peak electric demand was 11.2 kW for the 1979-1980 heating season. For 1980-1981, the heating season performance factor increased to 3.16 while the maximum peak electric demand decreased to 4.33 kW.

Johnson et al.[27] measured annual performance for the 1983-84 cooling and heating seasons of a ground-coupled heat pump system located in the TECH House I at the University of Tennessee. The ground-coupled heat pump system in the TECH House consists of 165 m^2 occupied single family residence. Design heat load of the house is 8.8 kW (heating degree-days for the Knoxville area is 1932 °Cdays) based on 18 °C. The heating capacity of the heat pump in use was 10 kW at T_{evap} = 2 °C, and T_{cond} = 30 °C. A ground coil pipe length of 213 m utilising 3 cm polybutylene pipe buried 1.2 m deep and laid in a horizontal serpantined array 1.8 m from one another were laid in parallel arrangement. They evaluated the performance of the horizontal ground coupled heat pump system in the heating mode for the period of November 1983 through March 1984 of total electric power consumed during the heating season, the compressor consumed 75 % of the total, the blower consumed 15 %, and the pumps 10 %. Table 2.2 shows seasonal performance summary of the in ground-coupled heat pump system for the heating season.

Table 2.2. Seasonal performance summary of the ground-coupled heat pump system for the heating season, Johnson at. al.[27]

Months	Power consp. (kWh)	Heat flows (kWh)	COP	PFi	PFo
November	581	1038	2.2	2.8	2.3
December	1310	2323	2.1	2.8	2.3
January	1346	2055	2.9	2.5	2.0
February	935	1432	1.9	2.5	2.0
March	970	1338	1.9	2.4	2.0
Heating season Total	5142	8186	2.0	2.6	2.1

Lund[28] described a model for determining the thermal performance of large-scale community solar heating systems with seasonal heat storage. Special attention was paid to the mathematical formulation of the storage unit comprising an uninsulated stratified hot water tank excavated in rock. The storage capacity of the surrounding ground may be utilised by vertical heat exchanger pipes. He prepared a computer program NORSOL in Fortran language for computing theoretical system performance of the system.

Lei [29] developed another model based on the finite difference method to simulate a vertical U-tube ground coupled heat exchanger connected to a heat pump system. Schematic diagram of their ground coupled heat pump system is shown in Figure 2.3. The two-dimensional cylindrical coordinate system was used in their model. The problem was converted from a three dimensional transient heat transfer problem. This problem was solved using the Crank-Nicolson implicit method on the near-tube region, where high degree of accuracy in simulation is required, and explicit method was used for all other points. A model was developed for this system and an experiment was also conducted to validate the model. Several simulation runs with different values of model parameters were conducted, and the results agreed with experimental data with only about 0.2 % to 5 % discrepancy for a 160 minute simulation period.

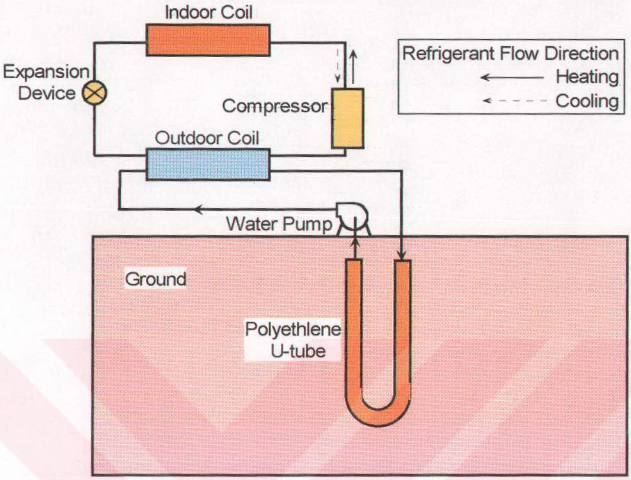


Figure 2.3. A water source ground coupled heat pump system, Lei[29]

Martin[30] determined the effects of individual design parameters on the performance of single-pipe, horizontal, ground-coupled heat pump systems. He built a computer model to predict the performance and energy consumption of a 10 kW residential heat pump system and compared the results to field data of two homes in Oklahoma. He also conducted an economic analysis to determine the economically optimum design. Their conclusions that may be drawn from this 10 kW residential heat pump system study are;

1. soil conductivity has a significant effect on the water temperature in the ground coil.
2. The economically optimum length of pipe for burying at any depth is between 305 m and 533 m.
3. There is no economic advantage to justify placing the ground coil deeper than 1.2 m.

4. There is no economic penalty for placing an extra 91 m in the ground coil beyond the calculated minimum coil length.

Ünsal[31] presented two solutions which can be utilised in seasonal thermal storage system simulation studies in cylindrical and spherical coordinates. He applied the complex finite Fourier transform techniques to transient heat conduction problems arising in the analysis seasonal thermal energy storage based heating systems as obtained two exact periodic solutions for the prediction of the transient temperature field during annually periodic operation of seasonal underground cylindrical and spherical thermal energy storage systems.

İnallı[32] investigated storage of solar energy in spherical and cylindrical tanks buried in the ground and thermal performance of a solar aided domestic heating system coupled with a heat pump. He gave computational procedure for finding water temperature and temperature distribution inside the geological structure surrounding the tank. He applied the complex finite Fourier transform technique used a suitable finite difference procedure to the dimensionless equations. He prepared a computer simulation program to obtain the annual periodic thermal performance of the heating system that consists of a building, solar collectors, heat storage tank and a heat pump. He presented results obtained from execution of the simulation program, and compared them with similar data available in the literature.

Tarnawski and Leong[33] prepared a computer program based on a one-dimensional mathematical model. They used the computer program to simulate a heat pump system coupled with a double-layer ground heat exchanger over a period of three or more years. At the end of the study, following conclusions were drawn:

1. An increased distance between pipes provides a higher seasonal performance factor (SPF) for a heating season.
2. The distance corresponding to the maximum value of the SPF varies generally with pipe spacing in the range of 4 to 4.5 meters

depending on the free ground area and the gradient of soil temperature.

3. A shorter distance between pipes, e.g., of 2 m, provides a smaller but more stable SPF over a longer period, even up to 10 years.

Lund and Östman[34] developed a three dimensional numerical model for seasonal heat storage in the ground using vertical pipes. A schematic diagram of their ground storage is shown in Figure 2.4.

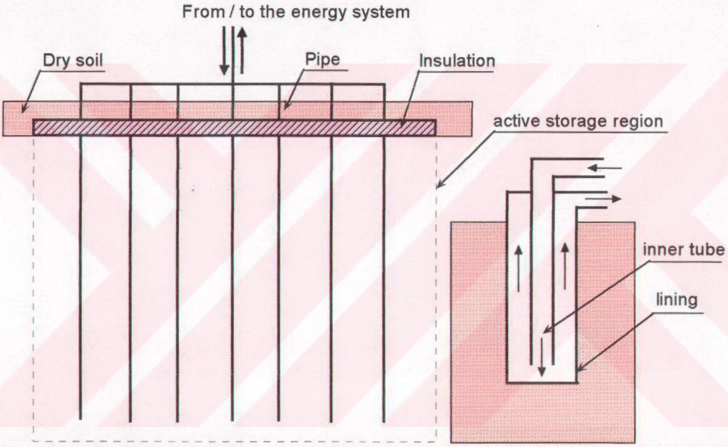


Figure 2.4. Schematic representation of a ground storage and of a single duct, Lund and Östman[34]

Their numerical model accounts for convective heat flows in the ground. The heat storage was employed in a district solar heating system with a heat pump. They studied the effects of storage volume, storage medium, collector area, and collector type on system performances for the Helsinki (60°). They obtained results for a 500 houses community with a collector area of 35 m^2 per house and a rock storage volume of 550 m^3 per house. This would provide a solar fraction of 70 %. Large storage volume reduced the solar collector requirement. The behaviour of the storage efficiency was found to depend on the volume and on the solar fraction requirement.

Brunström et al.[35] measured performance data of the Lyckebo project for a duration one year. The Lyckebo project is a large-scale solar district heating system with seasonal storage in a rock cavern. The solar district heating plant at Lyckebo, 13 km to the north of Uppsale, Sweden, has been in operation since the summer of 1983. The heating plant consists of highly efficient flat plate solar collectors of 4320 m^2, a rock cavern of 100000 m^3 for seasonal storage and a local distribution system. Schematic drawing of the Lyckebo system is given in Figure 2.5. They presented results for one year of operation of the plant, from April 1, 1984 to March 31, 1985. They discussed storage performance and annual energy storage efficiency of the system. They found that heat production of the collectors for this period was 1.24 Gwh, 294 kWh/m^2, at an energy-weighted mean operating temperature of 62 °C on the primary side of the heat exchanger. The corresponding global collector plane irradiation amounted to 989 kWh/m^2, resulting in a collector field efficiency of 29 per cent. The annual storage heat loss amounted to 3.14 Gwh, corresponding to 26 per cent of the total heat production. The annual mean flow-weighted loading temperature was 84 °C and the mean unloading temperature was 64 °C. They discussed the performance results of the Lyckebo system and decided that the solar district heating with seasonal storage in a rock cavern is a technically realistic alternative in the Swedish energy system.

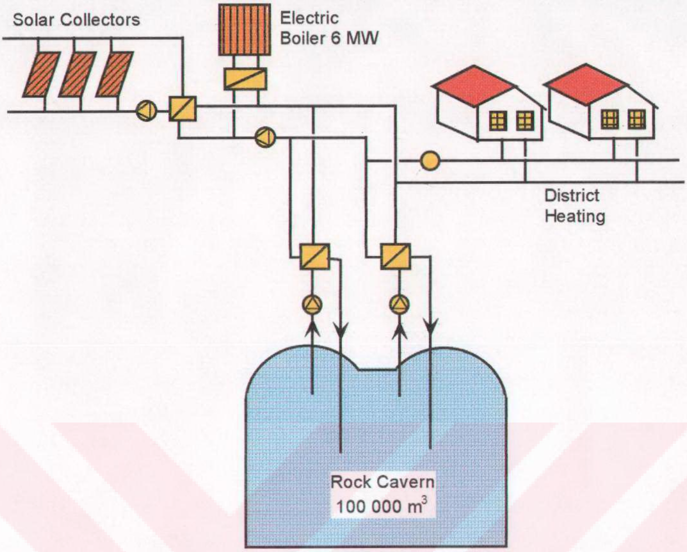


Figure 2.5. Schematic drawing of the Lyckebo system.

İnallı et al.[36] employed an analytical model for a domestic heating system assisted by solar energy stored in an underground spherical container. The heating system consists of solar collectors, an energy storage, a house and a heat pump. The analytical model employed calculates annually periodic water temperature in the storage vessel, and the temperature distribution in the surrounding ground outside the store. They computed annually periodic storage temperature, collector efficiency, performance coefficient of the heat pump and annual solar fraction by using their computer program, and presented the results in graphical form.

İnallı and Ünsal[37] analysed a solar assisted space system with a heat pump and a cylindrically shaped seasonal energy storage buried in earth. They transferred the transient temperature field problem to a form consisting of defined by a differential equation, boundary conditions and an energy balance for the storage. The transformed problem was solved using the finite difference method. The solution obtained was utilised to calculate annual performance of the space heating system. They calculated monthly house heat load and collector useful energy gains by using degree-day and $\bar{\phi}$ methods. They calculated the annually periodic storage temperature and energy fractions of the heating system as a function of earth types, storage volume, and depth of the store from the ground level. They also estimated effects of the geometric properties of the cylindrical tank, and thermal properties of the earth on the annual performance of the space heating system.

Hahne and Hornberger[38] installed a solar heating systems for an office building with laboratories and lecture rooms at Stuttgart University in 1985. Their solar heating system consists of 211 m^2 of unglazed solar collectors, a 1050 m^3 water- flooded pebble bed heat storage, heat pump, and a building with a space of 1375 m^2 for office and laboratories with a calculated heat demand of around 150 MWh per year. A Schematic picture of the heating system and cross-section of the underground energy storage are shown in Figure 2.6 and 2.7 respectively. The whole system has worked successfully for five years under varying conditions. In the first two heating periods, they found that about 60 percent of the heat demand could be covered by solar energy; but the yearly heat pump coefficient of performance (COP) was around 2.76. With an improved heat pump a monthly COP of 3.6 was obtained, and heat losses from the storage amounted to about 20 percent. Heat balance obtained for the store during two heating seasons is given in Table 2.3.

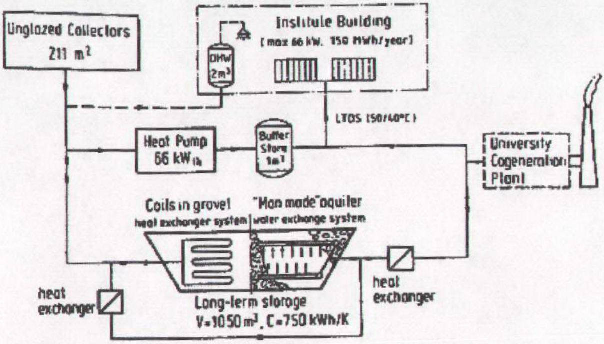


Figure 2.6. Schematic picture of the heating system, Hahne and Hornberger[38]

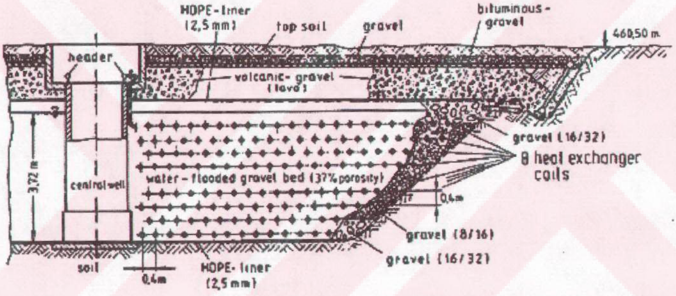


Figure 2.7. Cross-section of the underground energy storage, Hahne and Hornberger[38])

Table. 2.3. Heat balance of the store in two heating seasons, Hahne and Hornberger[38]

in MWh	Heat input	Heat extracted	Heat to Ground	Heat recovered	Heat lost	Differ. in stored heat	Storage efficiency
1986/87	64.4	56.0	27.4	15.7	11.7	3.3	82 %
1987/88	69.1	56.1	27.4	14.4	13.0	0	81 %

Shelton[39] studied thermal interaction between underground heat storages and surrounding ground. He presented and discussed analytic steady state solutions for hemispherical underground energy storage charged by solar collectors. Typical example of the assumed storage geometry and temperature profiles in the analytical portion of their study is shown in Figure 2.8. He considered one small and another large model system which are one and ten family solar assisted space heating systems. The systems are assumed to have heat load of 21000 and 210000 Btu/hr. He calculated the net heat loss to the ground during a 30 day period with a cold start. 21 % of collected heat was lost to the ground amounting to 10 % of the collected solar heat for a single house. He also calculated a 8.6 % loss from the energy storage to the ground and a 6.9 % gain from the ground to energy storage for ten houses. He estimated the fractions of total heat demand supplied by the solar to be between 40 and 69 percent for all cases. (steady-state and time dependent model)

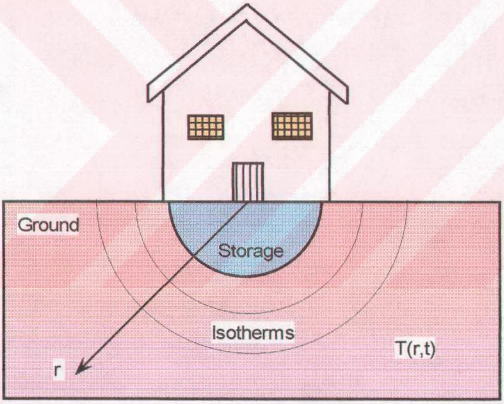


Figure 2.8. Typical example of the assumed storage geometry and temperature profiles, Shelton[39]

Ayhan et al.[40] installed an experimental setup to investigate the performance of a solar assisted heat pump space heating system with energy storage. The experimental setup consisted of 75 m^2 space (laboratory), 30 m^2 flat plate solar collectors, a heat pump, a water circulation pump, an energy storage and measuring devices. Calcium

Chloride Hexahydrate (CaCl26H2O) was used to store thermal energy obtained from the solar collectors. They obtained following results,

1. Heat pump COPH changes between 3 and 5 that depends on storage temperatures.
2. The system COPS is lower than the heat pump COPH.
3. Collector efficiency is about 70 % yearly except for December.
4. Air-to-air heat pump is not sufficient to keep the house at design the inside air temperature.

They pointed to the necessity for making economical optimisation to find out whether systems may be feasible or not in the black sea region of Turkey where the solar radiation is lower than that in Gaziantep.

Catan and Baxter[41] proposed to minimise the life cycle of a Ground Coupled Heat Pump (GCHP) by optimising design of both the heat pump and the ground coil. In order to achieve their objective, they developed several computer models to make performance and cost predictions for GCHP systems. By optimisation of a GCHP system, they determined the values of a limited set of design variables which yielded the lowest life cycle cost is predicted. The life-cycle cost of GCHP system (water-source heat pump with a horizontal ground coil) for an 167 m^2 house in Pittsburgh, was minimised in 7 years. Simple payback for the optimised GCHP system, relative to conventional air source heat pump, was under 3 years. The water-source heat pump package resulting from the optimisation was calculated to be 21% more expensive than its conventional counterpart. Heating coefficient of performance, on the other hand, is about 20% higher, than the conventional counterpart.

Metz[42] tested performance of an earth coil based ground coupled heat pump system that was operating since October 1980 in Upton, Long Islands, New York. The ground-coupled heat pump system was used to heat a test house which has a 104 m^2 floor area with 15 cm of fiberglass insulation in the attic. The heating load of the building was determined experimentally to be 7.8 Mj/°C day. Earth-coil consisted of 155 m of nominal 25.4-12.5 mm medium density polyethylene pipe buried 1.2 m deep. The pipe was laid in earth 16 m long and spaced about 2 m apart. The system has been operated with a heat transfer fluid consisting of water and 20% to 30% volumetric concentration of ethylene glycol. Operating temperatures of -7 °C and 40 °C for the earth-coil fluid were also imposed. The heating system described is shown in Figure.2.9.

He obtained heating performance of the ground coupled heat pump heating system for the 1981-82 heating season which is summarized in Table 2.4.

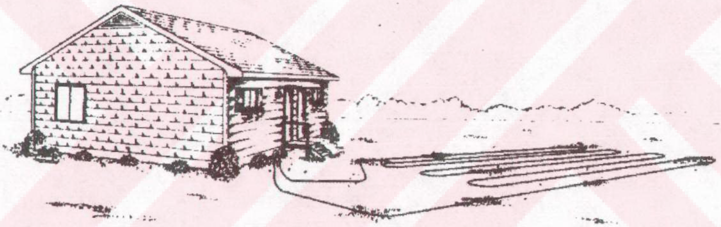


Figure 2.9. Earth coil layout with a test house, Metz[42]

Table 2.4. 1981-82 Seasonal system performance summary, Metz[42]

Average Far field temperature	+ 4.3 (°C)
Average earth coil temperature	+ 0.4 (°C)
Heat to house	+ 26.0×10^9 J
Heat to Ground	- 15.5×10^9 J
Heat pump energy	2683 kWh
Circulation pump energy	256 kWh
Heat pump COP	2.70
System COP (SPF)	2.46
Maximum daily U-value,	3.34 W/m°C

Mohammed-Zadeh et al.[43] modeled a general numerical model for analysis of ground loop heat exchangers for residential heat pump systems. They built a model from component modules, which were linked together to form a system. These modules include ground coupling of a pebble bed, a surface tank, a closed loop vertical well and a horizontal pipe system. A typical ground coupled heat pump system is shown in Figure 2.10. Their pebble bed storage is 13.7 m long, 1.2 m deep and 1.5 m wide. The surface tank is described to be 13.7 m long, 1.8 m wide and 0.46 m deep. The well heat exchanger consisted of an air insulated PVC pipe that carries the working fluid (water) from the heat pump to the bottom of the well. Two dimensional transient heat conduction equation was used to calculate the temperature fields surrounding the ground coupled heat exchanger. The transient conduction equation was solved by an explicit finite difference method. The predicted seasonal coefficient of performance for heating was compared with previously reported experimental results for the same system.

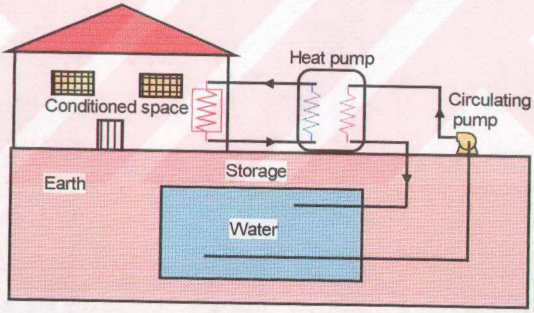


Figure 2.10. Typical ground coupled heat pump system, Mohammed-Zadeh et al.[43]

The experimental and predicted results obtained for three systems are given in Table 2.5 below. Their predictions are in good agreement with experimental results. They observed that coefficient of performance of a vertical well heat pump system was higher than type thermal storage a heat pump heating system with a pebble bed.

Table 2.5. Experimental and predicted results of the heating system, Mohammed-zadeh et al.[43]

Storage Types	Experimental COP	Predicted COP
Pebble bed	3.0	3.15
Surface tank	2.6	2.80
Well	3.5	3.40

Mei and Baxter[44] measured the seasonal performance of a ground coil heat pump system with six U-tubes of an equal depth that was installed at the test side in Knoxville, Tennessee. The heating system was operated for one cooling and one heating season. During that time, the cooling and heating seasonal performance factors were obtained as 1.75 and 2.85 respectively. An annual performance factor of 2.39 for the ground coil heat pump heating system was achieved, compared to 2.10-2.17 for an air source heat pump heating system. They found the hourly integrated peak demands for the ground coil heat pump system to be 3.3 kW for winter and 3.8 kW for summer, and peaks for the air source heat pump system were 9.9 kW and 3.7 kW, respectively.

Phettelace et al.[45] measured the performance of the ground-coupled heat pump system in a family housing unit. The heating system consisted of 10 ground-coupled water-to-air heat pump system, and a residential housing complex in Ft Polk, Louisiana. The ground-coupling heat exchanger consisted of two 10 cm diameter wells 61 m deep drilled adjacent to each residential unit. The performance of each of the 10 heat pump system was closely monitored. They monitored temperature at various depths of a test well at Ft. Polk as well as outdoor air temperatures at the same location. Sample soil and air temperature distributions monitored at Ft. Polk are shown in Figure 2.11. They also obtained as average COP of 3.5 by measuring the performance of the ground coupled heat pump system.

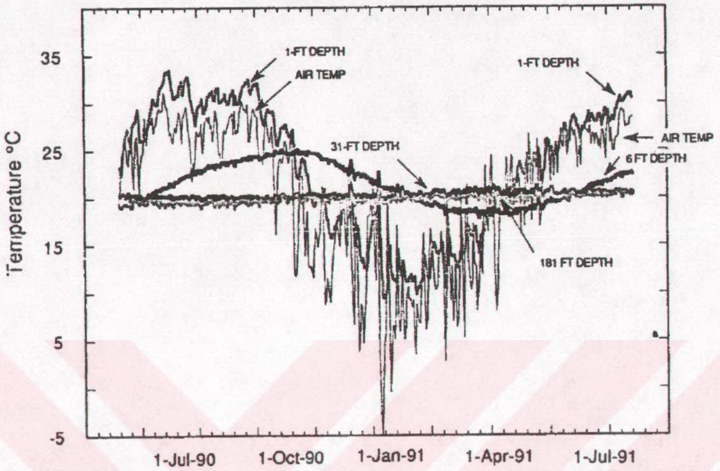


Figure 2.11. Sample soil and air temperatures at Ft. Polk.(air temperatures are daily averages from 2-hr scans; ground temperatures were taken at midnight each day.), Phettelace et al.[45]

Mei and Fisher[46] described an analytical and experimental study of a design of vertical, concentric-tube ground-coupled heat exchangers for use in heat pump applications. They developed a mathematical model and a computer program to simulate the operations of the ground-coupled heat exchanger. The experimental apparatus consisted of two concentric 47.2 m polyvinilcloride (PVC) pipes with associated instrumentation .

They obtained data using the experimental apparatus. The data was used to validate the computer program, and the computer model was then used to study the effects of variations in the heat exchanger length, diameter, flow rate, and thermal conductivity of the ground on the heat exchanger performance.

Their results for vertical, ground coupled heat exchangers were validated by laboratory tests. Conclusions of their work are summarized as:

1. Using high thermal conductivity materials will increase the amount of energy exchange between the heat exchanger.
2. Heat capacity of a heat exchanger increases with increasing flow rate of the working fluid, diameter of the outer casing, and length of the heat exchanger.

Edwards et al.[47] tested performance of a direct expansion heat pump system that is uniquely coupled to the earth through a Ground Loop Heat Exchanger Concentrator (GLHXC). The earth coupled heat exchanger consists of 31.8 m of 2.22 cm OD copper tubing that is immersed in a water bath. The water bath is formed by using a semi-cylindrical section of metal culvert. Gravel is filled around the culvert up to a depth of 0.91 m. Soil is placed on top of the gravel. A vapor liner is put in the excavated trench to retain the water.

The experimental results for the GLHXC system are given in Table 2.6 below for heating operations based on a monthly basis.

Table 2.6. Performance results for the GLHXC system, Edwards et al.[47]

Months	Average Heating COP	Average Daily Heating Effect kWh/day
December-1984	4.34	56.40
January-1985	3.91	45.96
February-1985	4.51	56.64

Kavanaugh[48] conducted tests to determine the acceptibility and operational characteristics of vertical ground coupled heat pump systems in southern climates. The heating system in his project consists of a residence with 150 m^2 floor area, a heating load of 13 kW, a desuperheater for domestic water heating with a capacity of approximately 1.5 kW, a water-to-air heat pump with a reciprocating compressor rated at 13 kW coupled to a 180 m vertical 2.5 cm polyethylene U-bend ground tube.

The water-to-air heat pump operated approximately 15 % below its rated capacity, he found that the system's heating seasonal performance factor was approximately 11.6 Btu/W.h (COP= 3.4) and COP was 3.3 (11.3 Btu/W.h) for the design day.

Johnson et al.[49] reported the performance of two different ground-coupled heat pump installations, for two single-family houses in a Tennessee experimental facility during the 1984-85 heating season. The first heating system is House I; 165 m^2 occupied area, single-family, frame construction with R-19 walls and R-22 ceiling insulation. Design heating load of house was 8.8 kW at -10 °C. House I system includes a water source, heating only heat pump rated at 10.56 kW.

Test results of the ground-coupled heat pump systems in House V and I are given in Table 2.7. Conclusions of the study are:

1. A comparison of the heat pump units exclusive of the ground coil showed that unit in House V has 6 % higher COP than the unit in House I in the heating mode.
2. The ground coil of the House I system is 1.2 m deep and back filled with sand, while the House V system is buried between 1.2 and 1.8 m depth and is backfilled with clay. During the winter of 1984-85, It was evident that the sand is created significant drop in performance due to its inability to dehydrate to a saturated condition during the winter months as does the clay.

Table 2.11. Performance results for the GLHXC system, Johnson et al.[49]

Period	SPFi	SPFo
Test House I	2.23	1.87
Test House V	2.84	2.56

Beck et al.[50] developed a new transient solution estimating the heat transfer around a single buried steam pipe. The transient heat conduction model was employed to describe the temperature distribution near the buried pipe. The physical model with a steam pipe located at a depth D below the soil surface is shown in Figure 2.12. The solution technique uses Green's functions and is particularly appropriate at distance greater than two or three pipe radii from the pipe. The model was used to design an experiment for determining buried steam pipe heat loss at different depths. They used the analytical solution to determine optimum locations for temperature measurements.

They plotted the temperature distribution in earth surrounding the steam pipe as a function of distance for fixed times as shown in Figure 2.13. They made the following observations:

1. The transients are quite slow. If a pipeline is not used during an extended period, then the temperature can take as long as ten months to approach steady state conditions after being reactivated.
2. The thermal effect can extend to a considerable distance from the pipe; 50 percent of the maximum steady state value is obtained at a distance of about 1.2 m (about D), and the distance to reduce to 10 percent of the maximum is reached at 3.6 m (about 3D).
3. The temperature changes considerably between $\alpha t/D^2 = 1$ and 10. The difference in the temperature at $x = 0$ and that at a moderate distance (such as 1.2 or 2.4 m) is nearly constant for $\alpha t/D^2 > 1$ and for $x < 2D$.

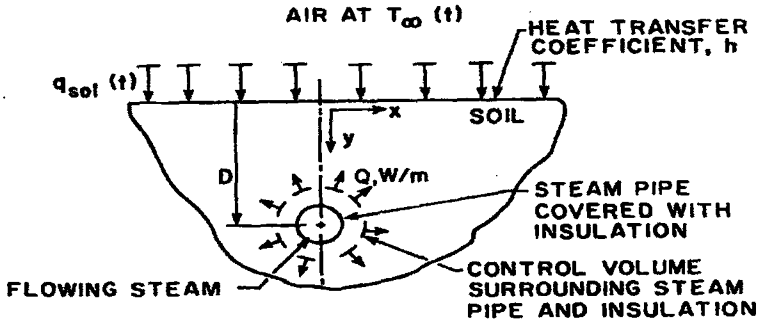


Figure 2.12. Diagram showing a steam pipe buried at a depth D below the soil surface, Beck et al.[50]

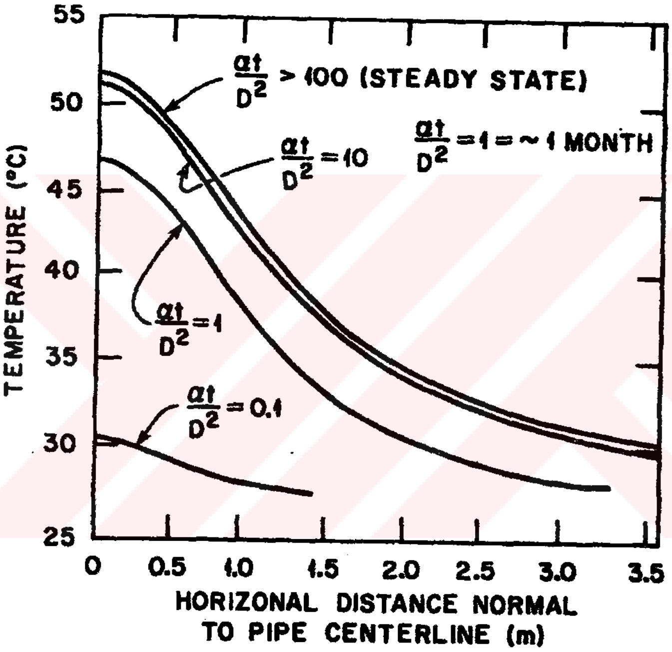


Figure 2.13. Temperature profiles at y=0.23 m for D=1.22 m, Q=385 W/m, Bi=10, $T_0+T_1(y,t)$=27.2 °C, Beck et al.[50]

Kavanaugh[51] has described a method of adopting the cylindrical heat source solution to predict thermal behavior of a conventional U-bend ground-coupling tubes. Ground-coupled heat pumps normally consist of a water-to-air heat pump, a closed-loop piping network buried in the soil, and an interconnecting piping loop. Heat is absorbed from the ground by the liquid inside the piping network. The heat pump delivers the heat to the building system. One of the more effective ground heat exchanger types was a vertical U-bend configuration as shown in Figure 2.14. Transient heat transfer problem in-ground cylinders has traditionally been modeled with either a numerical or a line source solution. He obtained performance results and compared the energy use of a ground-coupled system with conventional heating and cooling systems.

His results are given in Table 2.8. He concluded that; very little auxiliary heating was required because of the stable capacity of the ground-coupled heat pump system. Annual energy requirement was reduced by 40% compared to conventional single speed air source heat pump system and demand was reduced to 6 kW(56%) (in the heating mode).

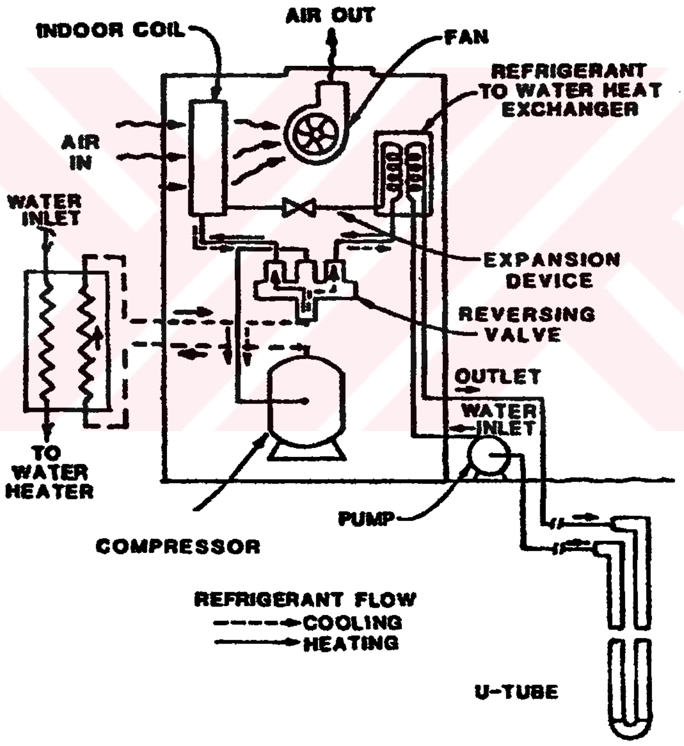


Figure 2. 14. Simple ground-coupled heat pump system, Kavanaugh[51]

Table 2.8. Performance results of air source and ground-coupled heat pump systems.

Energy need	Conventional air source heat pump	Ground-coupled heat pump
Heat pump heating (kWh)	6201	3498
Auxiliary heating (kWh)	536	12
Total heating (kWh)	6737	3510
Maximum heating demand kW	13.5	6

Mei[51] analyzed operation of a two-coil trench ground-coupled heat pump system for winter heating application. Schematic of the coil arrangement is shown in Figure 2.15-16. He presented a three-dimensional mathematical model based on energy conservation. His mathematical model accounts for fluid flow inside the coil, effects of coil material, coil size, and coil cyclic operation, as well as effects of coil thermal interference.

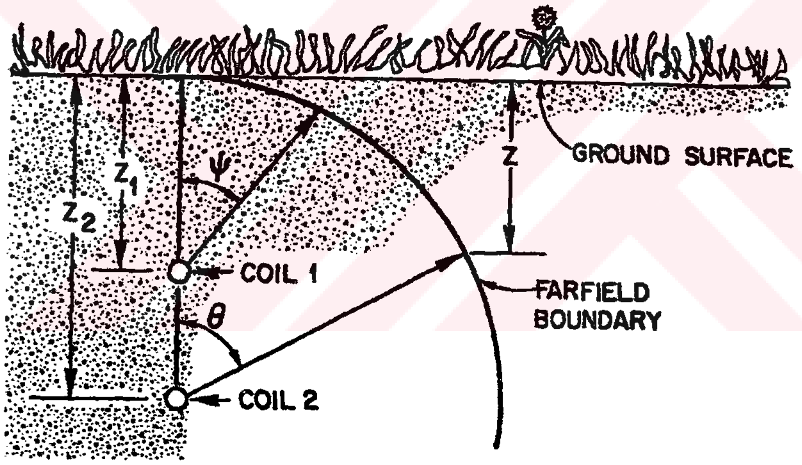


Figure 2.15. Schematic of two-coil arrangement, view 1, Mei[52]

A simulation run based on the University of Tennessee's field experimental data for 28 days, (by inputting the experimental ground coil inlet fluid temperature as a function of time) indicated excellent prediction of the test results for the energy exchange between the coil and ground. He also performed a parametric study of the effect of the fluid inlet position. The effect of thermal interference is shown in Figure 2.17. Comparison of measured and calculated daily energy absorbed from ground differed by a maximum of 27 percent, with the average error being than 12 percent.

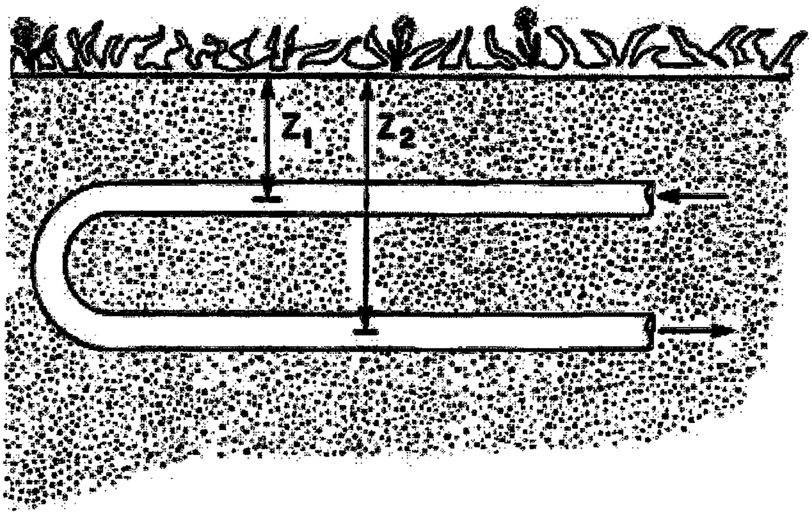


Figure 2.16. Schematic of two-coil arrangement, view 2, Mei[52]

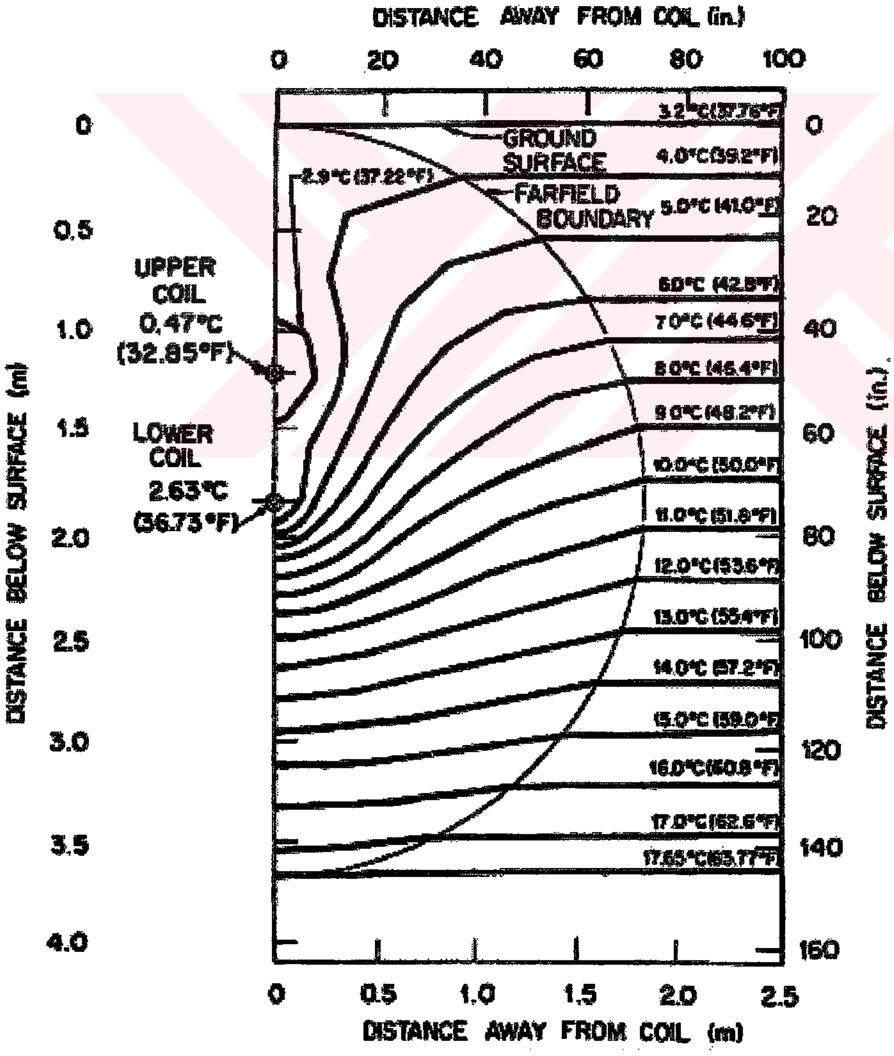


Figure 2.17. Calculated ground temperature distribution after 28-day coil operation.

CHAPTER III

MODELLING OF A SOLAR AIDED HEAT PUMP SPACE HEATING SYSTEM WITH A SEASONAL SPHERICAL UNDERGROUND THERMAL ENERGY STORAGE

3.1. INTRODUCTION

An analytical solution procedure is presented in this chapter for predicting the annual periodic performance of solar assisted space heating system with a seasonal spherical underground thermal energy storage and a heat pump. The space heating system consists of flat plate solar collectors, an underground spherical energy storage, a heat pump, and a house to be heated in winter season. The heating system under investigation is shown in Figure 3.1.

The solar collectors are mounted on the roof of the house and directed toward to the south. The collectors absorb solar energy over the whole year. The absorbed energy is transferred to the water in the underground thermal energy storage throughout the whole year by circulating antifreeze-water solution in a closed cycle system. The stored energy is extracted by the heat pump from the thermal energy storage for space heating only during the heating season. The heat pump operates only when the temperature of the storage is not sufficient to keep the house at design inside air temperature. The heat extracted by the heat pump is supplied to the house by panel type heat exchangers (or by fan coil units).

The total annual energy supplied to the heating system during the whole year is the sum of the solar energy extracted from the storage and heat pump work. A fraction of the solar energy will be lost into the earth surrounding the storage and the remaining part will be used for house heating during the winter season. An analytical calculation procedure is given in the following sections. Temperature of the storage and geological structure surrounding the storage, heat pump work, available solar energy gain rate, and house heat load are to be determined from the analytical model for the heating system.

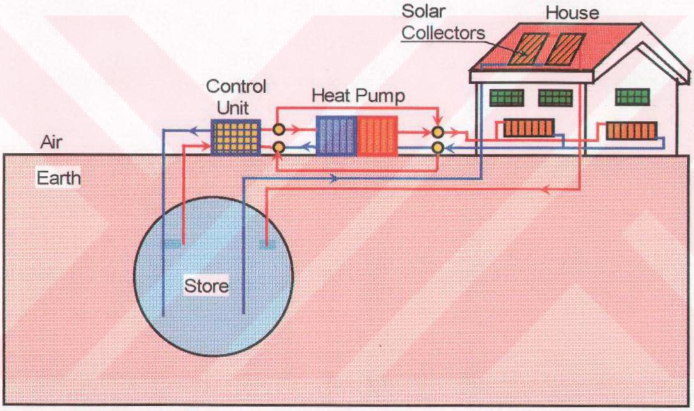


Figure 3.1. Heat pump system with a spherical storage in deep earth

The underground thermal energy storage is an the essential part of the heating system under investigation. Analytical solution for the annually periodic transient temperature field outside the thermal energy storage is to be obtained and used in the present simulation. The problem formulation consists of the partial differential equation, initial and boundary conditions and an energy balance for the storage. The problem formulation is put into dimensionless form using dimensionless variables. The dimensionless formulation of the problem is solved by an application of complex finite Fourier transform technique which yields another problem for the transformed temperature. Net heat input rate to the storage is equal to difference between the energy input by the solar collectors and heat extracted from the storage by the heat pump. Useful solar energy for each month is estimated using the $\bar{\phi}$ method described by Duffie and Backman (1980).

3. 2. ESTIMATION OF THE TEMPERATURE FIELD OUTSIDE THE THERMAL ENERGY STORAGE

Figure 3.2 shows the schematic representation of the energy storage buried in deep earth. The following assumptions have been made to simplify the analysis: The spherical thermal energy storage is assumed to be located deep enough so that the far field ground temperature away from the storage is taken to be constant with respect to time and equal to the deep ground temperature T_∞; The geological structure around the energy storage is assumed to be homogeneous and having constant thermal properties in everywhere; The temperature in earth is considered to be a function of the radial coordinate and time $T(r,t)$. The water in the storage is assumed fully mixed and at a spatially lumped time varying temperature $T_w(t)$.

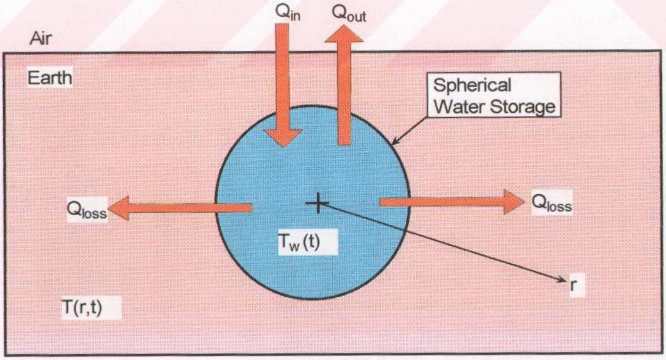


Figure 3.2. Schematic representation of the underground thermal energy storage

The annually periodic transient heat transfer problem outside the spherical thermal energy storage may be expressed in the spherical coordinate system as follows.

$$\frac{\partial^2 T}{\partial r^2}+\frac{2}{r}\frac{\partial T}{\partial r}=\frac{1}{\alpha}\frac{\partial T}{\partial t} \tag{3. 1}$$

$$T(a, t) = T_w(t) \tag{3. 2}$$

$$T(\infty, t) = T_\infty \tag{3. 3}$$

$$T(r, 0) = T(r, \text{one year}) \tag{3.4}$$

Where, α in equation (3.1), is the thermal diffusivity of the earth surrounding the thermal energy storage. The energy balance relating energy charge rate, energy accumulation rate in the storage, and the conduction heat loss rate into the surrounding earth is expressed in the following form.

$$Q=\rho_W c_W V_{WS}\frac{dT_W}{dt}-k A_S\frac{\partial T}{\partial r}(a,t) \tag{3. 5}$$

Where, Q is net energy charge rate to the storage. ρ_w, c_w and V_{ws} are density of water in the storage, specific heat of water and volume of the water in the storage respectively. k, a and A_s are the thermal conductivity of the soil, radius of the storage and surface area of the storage respectively. The transient heat transfer problem consisting of Equations (3.1)-(3.5) can be transferred into dimensionless form using the following dimensionless variables.

$$x=\frac{r}{a}\,,\qquad \tau=\frac{\alpha t}{a^2}\,,\qquad \phi=\frac{T-T_\infty}{T_\infty}\,,\qquad \phi_w=\frac{T_w-T_\infty}{T_\infty}\,,\qquad \phi_a=\frac{T_a-T_\infty}{T_\infty}$$

$$\psi(x,\tau)=x\,\phi(x,\tau)\,,\;\; Y=\frac{\alpha(\text{one year})}{a^2}\,,\;\; q=\frac{Q}{4\pi a k T_\infty}\,,\;\; p_s=\frac{\rho_w c_w}{3\rho c} \tag{3. 6}$$

Where, ρc given in equation (3.6) is the product of density and specific heat of soil surrounding the storage. The resulting dimensionless form of the problem formulation is given by:

$$\frac{\partial^2\psi}{\partial x^2}=\frac{\partial\psi}{\partial\tau} \tag{3. 7}$$

$$\psi(1,\,\tau)=\phi_w(\tau) \tag{3. 8}$$

$$\psi(\infty,\,\tau)=0 \tag{3. 9}$$

$$\psi(x,\,0\,)=\psi(x,\,Y) \tag{3. 10}$$

$$q=p_s\,\frac{d\phi_w}{d\tau}-\frac{\partial\psi}{\partial x}(1,\tau)+\psi(1,\tau) \tag{3. 11}$$

The dimensionless problem consisting of equations (3.7)-(3.11) is transformed by an application of the Complex Finite Fourier Transform defined by:

$$\psi(x,\tau)=\sum_{n=-\infty}^{+\infty}\psi_n(x)\exp\left\{i\frac{2\pi n\tau}{Y}\right\} \tag{3. 12}$$

$$\psi_n(x)=\frac{1}{Y}\int_{-Y/2}^{+Y/2}\psi(x,\tau)\,\exp\left\{i\frac{2\pi n\tau}{Y}\right\}d\tau \tag{3. 13}$$

When the transformation given by (3.12)-(3.13) is applied to the dimensionless problem formulation defined by (3.7)-(3.11), the following is obtained.

$$\frac{d\psi_n}{dx^2} - \frac{i2\pi n}{Y}\psi_n = 0 \tag{3. 14}$$

$$\psi_n(1) = \phi_{wn} \tag{3. 15}$$

$$\psi_n(\infty) = 0 \tag{3. 16}$$

$$q_n = p_s \frac{i2\pi n}{Y}\phi_{wn} - \frac{d\psi_n}{dx}(1) + \psi_n(1) \tag{3. 17}$$

The solution of the transformed dimensionless problem given by (3.14)-(3.17) is given by:

$$\psi_n(x) = \phi_{wn} \exp\left\{-(1+i)\frac{\sqrt{n\pi}(x-1)}{\sqrt{Y}}\right\} \tag{3. 18}$$

where

$$\phi_{wn} = \frac{q_n(\eta_1 - i\eta_2)}{\eta_1^2 + \eta_2^2} \tag{3. 19}$$

and

$$\eta_1 = 1 + \frac{\sqrt{n\pi}}{\sqrt{Y}} \qquad \eta_2 = \frac{\sqrt{n\pi}}{\sqrt{Y}} + p_s\frac{2\pi n}{Y} \tag{3. 20}$$

q_n, in equation (3.19), represents the Complex Finite Fourier Transform of the net dimensionless heat input rate to the storage. Estimation procedure of the q_n is given by the following section.

3.3. COMPLEX FINITE FOURIER TRANSFORM OF THE HEAT INPUT RATE

Available solar energy is charged by the flat plate solar collectors to the energy storage over the whole year. Part of the stored energy is supplied to the house during heating season by a heat pump. Dimensionless heat input rate into the energy storage $q(\tau)$ is equal to the difference between charged energy to the storage and energy extracted by the heat pump, which may be expressed as:

$$q(\tau) = q_u(\tau) - q_{hp}(\tau) \tag{3. 21}$$

$q_{hp}(\tau)$ in Equation (3.21) is the energy extracted from the storage by the heat pump, which may be expressed as:

$$q_{hp}(\tau) = q_h(\tau) - \frac{w(\tau)}{\gamma} \tag{3.22}$$

When equation (3.22) is substituted into the Equation (3.21), net energy input rate to the energy storage becomes:

$$q(\tau) = q_u(\tau) - q_h(\tau) + \frac{w(\tau)}{\gamma} \tag{3.23}$$

Where $q_h(\tau)$ and $w(\tau)$ are dimensionless house heat load and heat pump work respectively. Calculation procedures for $q_h(\tau)$ and $w(\tau)$ are explained in section 3.4. $q_u(\tau)$ in Equation (3.23) is the dimensionless energy collection by the solar collectors. The method of calculation of available solar energy gain is based on the $\bar{\phi}$ method and is described in section 3.5.

Annual dimensionless temperature distribution in earth given by Equation (3.12) depends on the complex Fourier coefficients of the dimensionless net heat input rate, $q(\tau)$ given in Equation (3.23). $q(\tau)$ may be expressed by a Fourier series with a period of one year. Let us define g_i to be monthly components of the dimensionless energy input rate to the storage. The dimensionless heat input rate, $q(\tau)$ may be expressed by the following vector.

$$q(\tau)=\left(g_1, g_2, g_3, g_4, g_5, g_6, g_7, g_8, g_9, g_{10}, g_{11}, g_{12}\right) \tag{3.24}$$

The dimensionless heat input rate, $q(\tau)$ given by Equation (3.24) can be expressed by the following Fourier series with a period of one year.

$$q(\tau)=\sum_{n=-\infty}^{+\infty} q_k\, e^{i\nu_k\tau} \tag{3.25}$$

Complex Fourier coefficients of the dimensionless heat gain rate, q_k, in Equation (3.25) are given by:

$$q_0=\frac{1}{12}\sum_{j=1}^{12} q_j \qquad \text{for} \quad k=0 \tag{3.26}$$

$$q_k=\frac{1}{2\pi k}\sum_{j=1}^{12} q_j\left\{\eta_{3,j}+i\,\eta_{4,j}\right\} \qquad \text{for} \quad k\geq 1 \tag{3.27}$$

Where

$$\eta_{3,j} = \sin(2\pi k r_j) - \sin(2\pi k r_{j-1}) \quad (3.28)$$

$$\eta_{4,j} = \cos(2\pi k r_j) - \cos(2\pi k r_{j-1}) \quad (3.\ 29)$$

3.4 MODELLING OF THE HEAT PUMP

Heat pumps provide a means for reducing auxiliary heating requirements in a house with solar space heating system. It is a device which uses mechanical energy to transfer thermal energy from a source at lower temperature to a sink at a higher temperature. It is this feature that makes them attractive for application in domestic space heating and solar aided heating systems in particular. Heat pumps are often considered as a conventional heating system component rather than an active solar system component. Four basic heat pump system designs employed in space heating are:

1- Air as the heat source and air as the heating medium (air-to air).
2- Air as the heat source and water as the heating medium (air-to water).
3- Water as the heat source and air as the heating medium (water-to-air).
4- Water as the heat source and water as the heating medium (water-to-water)

Heat pumps have a large potential for providing energy conservation in heating systems. According to Calm[53], the air source heat pump is most commonly used, but it operates with very low efficiency during extremely cold weather and can cause an undesirable load situation for the electricity supplier. The ground coupled heat pump could maintain better performance during extremely cold seasons and could be a solution to this problem.

The ground coupled heat pump extracts heat from an underground energy storage source and supply heat to a house known as a sink at high temperature. A ground-coupled heat pump can greatly improve the overall system performance because of the stable year round deep ground temperatures. Water may be used for heat extraction from sources and heat rejection to sinks. Schematic representation of a ground coupled water-to-water heat pump used in the solar aided heating system understudy is shown in Figure 3.3.

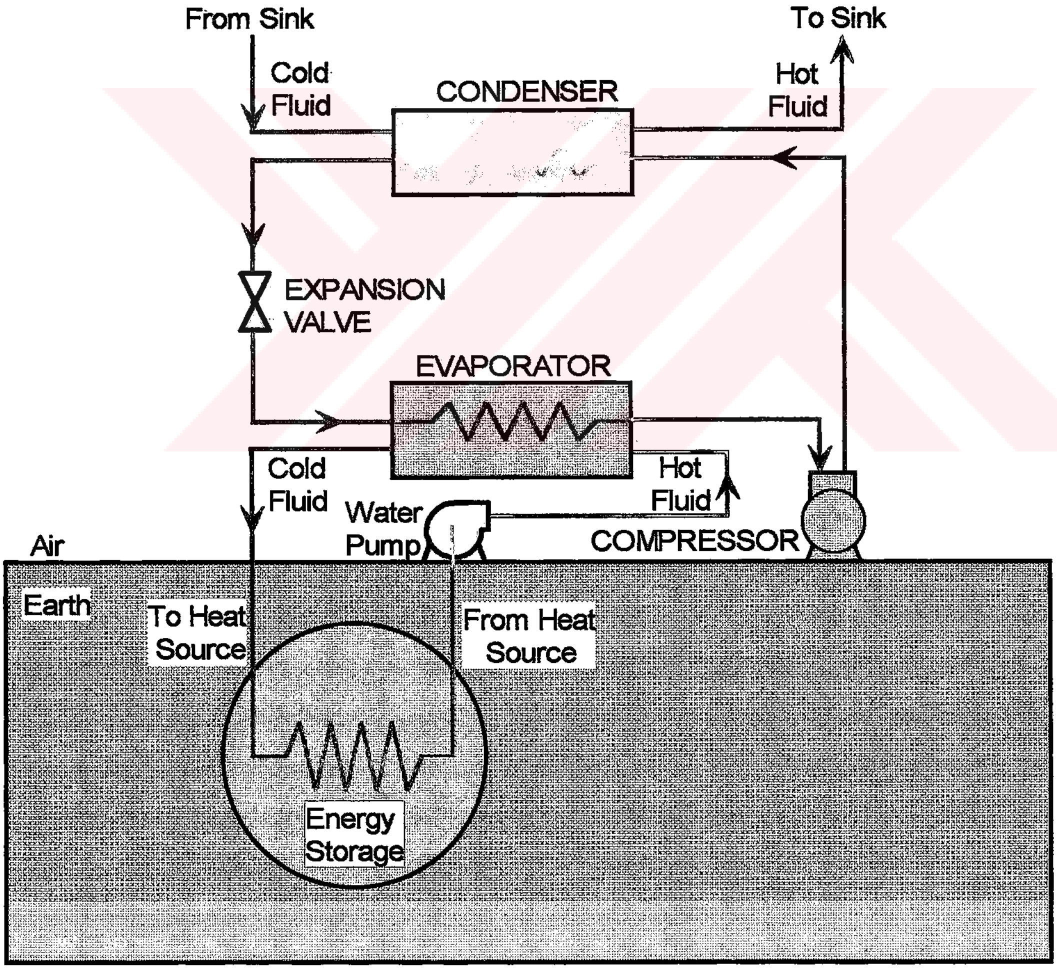


Figure 3.3. Schematic representation of the water to water heat pump.

An algebraic expression for heat pump simulation is given in this section. It is used to calculate dimensionless heat pump work. Coefficient of performance of heat pump, COP is a function of condenser and evaporator temperatures. Characteristic heat pump performance curve was adapted from experimental data of an actual conventional heat pump in this study. A very good fit to the experimental data was obtained from by the following expression as:

$$COP = \beta_{hp}\left(\frac{T_w + 100}{70} Ln\left[\frac{T_h}{T_h - T_w}\right] + \frac{35 - T_h}{40}\right) \tag{3.30}$$

β_{hp} is a coefficient which may vary depending on model and manufacturer. It is assumed equal to unity in this study. T_h is the temperature of the water in the load side heat exchanger. Monthly average energy requirement of the house in the heating season may be expressed as:

$$Q_h = (UA)_h(T_i - \overline{T}_a) \tag{3.31}$$

where, $(UA)_h$ is the UA-value of the house, T_i and $\overline{T}_a$ are the inside design air temperature and monthly average ambient temperature. The energy requirement of the space, Q_h is supplied by the heat pump via the load side heat exchangers which may also be expressed by:

$$Q_h = (UA)_{he}(T_h - T_i) \tag{3.32}$$

$(UA)_{he}$ in equation (3.32) is UA-value for the load side heat exchanger in the space. Heat supplied to the space can also be expressed by the product of the coefficient of performance of the heat pump and heat pump work:

$$Q_h = W(COP) \tag{3.33}$$

If the COP expressed in Equation (3.30) is substituted in Equation (3.33), The following equation is obtained.

$$Q_h = W\beta_{hp}\left\{\frac{T_w+100}{70}\,Ln\left[\frac{T_h}{T_h-T_w}\right]+\frac{35-T_h}{40}\right\} \tag{3. 34}$$

Q_h may be eliminated from equation (3.34) by substitution from Equation (3.31). Combining Equations (3.31) and (3.32), solving the resulting expression for T_h and substitution of the resulting expression into Equation (3.34), yields the following expression for the dimensionless work required by the heat pump.

$$w=\frac{\phi_i-\phi_a}{\beta_{hp}\left\{\left[\frac{T_\infty(\phi_w+1)+100}{70}\right]+Ln\left[\frac{T_\infty(\varphi+1)+273}{T_\infty(\varphi-\phi_w)}\right]+\frac{35-T_\infty(\varphi+1)}{40}\right\}} \tag{3.35}$$

where;

$$\varphi = u\left(\phi_i-\phi_a\right)+\phi_i \tag{3.36}$$

The symbols are defined as: ϕ_a, dimensionless outside air temperature; ϕ_i, dimensionless inside design air temperature; ϕ_w, dimensionless thermal energy storage temperature. The ratio of (UA)-value for the house to (UA)-value for load side heat exchanger given by:

$$u=\frac{(UA)_h}{(UA)_{he}}=\frac{T_h-T_i}{T_i-T_a} \tag{3.37}$$

$(UA)_h$ given in Equation (3.37) is equal to ratio of design heat load, Q_d of the house to design temperature difference.

$$(UA)_h = \frac{Q_d}{T_i - T_d} \tag{3.38}$$

3.5. ESTIMATION OF MONTHLY AVERAGE DAILY USEFUL ENERGY COLLECTION

The monthly average daily useful energy collection Q_u, is equal to the monthly component of the daily energy input to the storage. It can be calculated from the following formula reported by Klein[54] and Mitchel et al.[55]:

$$Q_u = A_c F_R \overline{(\tau\alpha)} \bar{H}_T \bar{\phi} \tag{3.39}$$

Where A_c is the collector surface area, and F_R is the collector heat removal factor. $\overline{(\tau\alpha)}$ is the monthly average transmittance absorbtance product for the collector. Its calculation method is given in Section 3.6. $\bar{H}_T$ in Equation (3.39) is the monthly average daily total solar radiation on a tilted surface. It is given by the following equation.

$$\bar{H}_T = \bar{R}\,\bar{H} \tag{3.40}$$

Where $\bar{H}$ is the monthly average daily total solar radiation on a horizontal surface. Calculation method of $\bar{R}$ is explained in the following section.

3.5.1. CALCULATION OF $\bar{R}$

$\bar{R}$ is the ratio of the monthly average daily total radiation on a tilted surface to that on a horizontal surface. $\bar{R}$ can be estimated by considering the beam, diffuse, and ground-reflected components of radiation and can be expressed as:

$$\bar{R}=\left(1-\frac{\bar{H}_d}{\bar{H}}\right)\bar{R}_b+\left(\frac{\bar{H}_d}{\bar{H}}\right)\frac{1+\cos\beta}{2}+\rho_g\left(\frac{1-\cos\beta}{2}\right) \tag{3.41}$$

$\bar{H}_d/\bar{H}$ in Equation (3.41) is the ratio of monthly average daily diffuse radiation to total daily radiation on a horizontal surface. Collares-Pereira and Rabl[56] developed the following correlation for calculation of $\bar{H}_d/\bar{H}$ as a function of monthly average clearness index, $\bar{K}_T$ and sunset hour angle w_s:

$$\begin{aligned}\frac{\bar{H}_d}{\bar{H}}&=0.775+0.00653(w_s-90)\\&\quad-\left[0.505+0.00455(w_s-90)\right]\cos(115\bar{K}_T-103)\end{aligned} \tag{3.42}$$

where

$$w_s=\cos^{-1}(-\tan\phi_L\tan\delta) \tag{3.43}$$

and δ is the solar declination angle calculated from:

$$\delta=23.45\sin\left(360\frac{284+n_d}{365}\right) \tag{3.44}$$

$\bar{R}_b$ in Equation (3.41) is the monthly average daily beam radiation ratio. Lui and Jordan[57] suggest that it can be estimated from the ratio of extraterrestrial radiation on tilted surface to that on a horizontal surface for a month. For surfaces directed due south, Lui and Jordan give the following equation for $\overline{R}_b$

$$\overline{R}_b = \frac{\cos(\phi_L - \beta)\cos\delta \sin w_s' + (\pi/180) w_s' \sin(\phi_L - \beta)\sin\delta}{\cos\phi_L \cos\delta \sin w_s + (\pi/180) w_s \sin\phi_L \sin\delta} \qquad (3.45)$$

Where w_s' is the sunset hour angle for a tilted surface given by:

$$w_s' = \mathrm{Min}\begin{bmatrix} \cos^{-1}(-\tan\phi_L \tan\delta) \\ \\ \cos^{-1}(-\tan(\phi_L - \beta)\tan\delta) \end{bmatrix} \qquad (3.46)$$

ρ_g in Equation (3.41) is the ground reflectance. Lui and Jordan[57] suggest that ρ_g varies from 0.2 to 0.7 depending upon the extent of snow cover.

3.5.2. THE UTILIZABILITY METHOD

The concept of utilizability has been pioneered by Whillier[58], and later generalised by Lui and Jordan[59]. It has been simplified and extended by Klein and Backman[60] who developed a general design method for solar systems involving flat plate collectors.

$\overline{\phi}$ given in equation (3.39) is the monthly average daily utilizability. It is defined as the sum for a month, over all hours and all days, of the radiation on a tilted surface that is above a critical level, divided by the monthly radiation. The value of $\overline{\phi}$ for a monthly period depends upon the distribution of daily total solar radiation, collector orientation, location, and time of the year. A correlation for $\overline{\phi}$ was developed by Klein[54] as a function of the monthly clearness index $\overline{K}_T$, a dimensionless critical radiation level $\overline{X}_c$, and geometry factor, $\overline{R}/R_n$. The resulting correlation is expressed as:

$$\overline{\phi} = \exp\left\{\left[A + B\left(\frac{R_n}{\overline{R}}\right)\right](\overline{X}_c + C\overline{X}_c^2)\right\} \tag{3.47}$$

where

$$A = 2.943 - 9.271\overline{K}_T + 4.031\overline{K}_T^2 \tag{3.48}$$

$$B = -4.343 + 8.853\overline{K}_T - 3.602\overline{K}_T^2 \tag{3.49}$$

$$C = -0.17 - 0.306\overline{K}_T + 2.936\overline{K}_T^2 \tag{3.50}$$

and the monthly average critical radiation ratio given in Equation (3.47), $\overline{X}_c$ is given by:

$$\overline{X}_c = \frac{H_{t,c}}{r_{t,n}R_n\overline{H}} \tag{3.51}$$

or

$$\overline{X}_c = \frac{1}{r_{t,n}R_n\overline{K}_T\overline{H}_0}\left[\frac{F_RU_c(T_{ci} - \overline{T}_a)}{F_R(\overline{\tau\alpha})}\right] \tag{3.52}$$

where the symbols are defined as; U_c, collector heat loss coefficient; T_{ci}, inlet fluid temperature to collector; $\overline{T}_a$, monthly average outside air temperature; $(\overline{\tau\alpha})$, monthly average transmittance-absorbtance product, $\overline{H}_0$, is the monthly average daily extraterrestrial radiation on a horizontal surface and is given by:

$$\bar{H}_0 = \frac{24*3600*G_s}{\pi}\left[1+0.033\cos\left(\frac{360n}{365}\right)\right]\left[\cos\phi_L\cos\delta\sin w_s + \frac{2\pi w_s}{360}\sin\phi_L\sin\delta\right] \tag{3.53}$$

The method of calculation for R_n defined in Equation (3.47) is given in next Section.

3.5.3. CALCULATION OF R_n

R_n given in Equation (3.47) is the ratio of radiation on a tilted surface to that on a horizontal surface at noon for an average day of the month. It can be calculated by considering the beam, diffuse and ground-reflected component of the radiation.

$$R_n = \left(1-\frac{r_{d,n}}{r_{t,n}}\frac{H_d}{H}\right)R_{b,n} + \left(\frac{r_{d,n}}{r_{t,n}}\frac{H_d}{H}\right)\left(\frac{1+\cos\beta}{2}\right) + \rho_g\left(\frac{1-\cos\beta}{2}\right) \tag{3.54}$$

where $r_{d,n}$ is the ratio of the diffuse radiation at noon to daily total radiation. The $r_{d,n}$, as suggested by Lui and Jordon[61], is given as:

$$r_{d,n} = \frac{\pi}{24}\frac{\cos w - \cos w_s}{\sin w_s - (2\pi w_s/3600)\cos w_s} \tag{3.55}$$

and $r_{t,n}$ is the ratio of the radiation at noon to daily total radiation which is given by the following equation given by Collares-Pereira and Rabl[56].

$$r_{t,n} = \frac{\pi}{24}(a_1 + b_1\cos w)\frac{\cos w - \cos w_s}{\sin w_s - (2\pi w_s/3600)\cos w_s} \tag{3.56}$$

where a_1 and b_1 depend on the sunset hour angle and are given by:

$$a_1 = 0.409 + 0.5016\sin(w_s - 60) \tag{3.57}$$

$$b_1 = 0.6609 + 0.4767\sin(w_s - 60) \tag{3.58}$$

$R_{b,n}$ in Equation (3.54) is the ratio of beam radiation on a tilted surface to that on a horizontal surface at noon. For surfaces facing directly towards the equator, $R_{b,n}$ can be expressed as Klein[54]:

$$R_{b,n} = \frac{\cos(\phi_L - \beta)\cos\delta + \sin(\phi_L - \beta)\sin\delta}{\cos\phi_L \cos\delta + \sin\phi_L \sin\delta} \tag{3.59}$$

H_d/H in Equation (3.54) is the daily diffuse radiation fraction. Collares-Pereira and Rabl[56] recommended a correlation for estimating H_d/H as a function of K_T. K_T is the ratio of the daily total radiation on a horizontal surface to the daily extraterrestrial radiation. For the average day considered in this analysis K_T was taken to be the same value of $\overline{K}_T$. According to Duffie and Backman[62]:

$$\frac{H_d}{H} = \begin{Bmatrix} = 0.99 & \text{for} \quad K_T \le 0.17 \\ = 1.188 - 2.272K_T + 9.473K_T^2 & \\ \qquad -21.863K_T^3 + 14.648K_T^4 & \text{for} \quad 0.17 < K_T < 0.75 \\ = -0.54K_T + 0.632 & \text{for} \quad 0.75 < K_T < 0.80 \\ = 0.2 & \text{for} \quad K_T \ge 0.80 \end{Bmatrix} \tag{3.60}$$

3.6. ESTIMATION OF THE MONTHLY AVERAGE TRANSMITTANCE ABSORBTANCE PRODUCT

The $\overline{\phi}$ design method, is given in Section 3.5. This method requires the monthly average transmittance absorptance product, $(\overline{\tau\alpha})$ as input data.

In this section, a method of estimating the monthly average transmittance-absorptance product is described.

Transmittance of a transparent collector cover, τ, and absorbtance of the collector plate, α, are functions of the material and the incidence angle of solar radiation. Beam, diffuse and ground-reflected components of the solar radiation are incident on the collector surface at different angles. The variation of incident angle of solar radiation with respect to transmittance absorptance product, $(\tau\alpha)$ must be known to calculate monthly average transmittance absorptance product of flat plate collectors. The $(\tau\alpha)$ can be calculated theoretically, but calculation of the product may be realistic using experimental collector parameters. Simon[63] presented performance test results for 23 different collector parameters. The for four different collectors widely used in applications are given in Section 3.8.3.

An expression for transmittance absorptance product as given by Simon[63] (as a function of incident angle of beam radiation) is:.

$$\frac{(\tau\alpha)}{(\tau\alpha)_n} = 1 + b_0\left(1 - \frac{1}{\cos\theta_i}\right) \tag{3.61}$$

$(\tau\alpha)$ at a given instant of time must be calculated as the radiation weighted average of $(\tau\alpha)$ for the beam, diffuse and ground reflected radiation components. which is given by the following equations:

$$(\tau\alpha) = \frac{H_b R_b (\tau\alpha)_b + H_d \frac{1+\cos\beta}{2}(\tau\alpha)_d + \rho H \frac{1-\cos\beta}{2}(\tau\alpha)_r}{H_T} \tag{3.62}$$

where, H_T is the solar flux on a tilted collector surface; H is the solar flux on a horizontal surface; H_b and H_d are the solar fluxes of beam and diffuse radiation on a horizontal surface respectively; $(\tau\alpha)_b$, $(\tau\alpha)_d$ and $(\tau\alpha)_r$ are the transmittance absorptance products for beam, diffuse and reflected solar radiation.

The Equation (3.62) provides a relationship for $(\tau\alpha)$ at a given instant of time. However, it is the monthly average value of $(\tau\alpha)$ which is needed for calculating the monthly average useful solar energy gain. Monthly average transmittance-absorbtance product is obtained by integrating of Equation (3.62) with respect to time over a month of N days. $(\overline{\tau\alpha})$, is given as:

$$(\overline{\tau\alpha})=\frac{\left(1-\frac{\overline{H}_d}{\overline{H}}\right)\overline{R}_b(\overline{\tau\alpha})_b+\frac{\overline{H}_d}{\overline{H}}\left(\frac{1+\cos\beta}{2}\right)(\overline{\tau\alpha})_d+\rho\left(\frac{1-\cos\beta}{2}\right)(\overline{\tau\alpha})_r}{\overline{R}} \qquad (3.63)$$

The calculation method of $\overline{R}$ and $\overline{R}_b$ defined in equation (3.63) is given in Section (3.5.1). $(\overline{\tau\alpha})_d$, in Equation (3.63), is calculated from the equation (3.61) for $\theta_d = \pi/3$, and $(\overline{\tau\alpha})_r$ is calculated from the following equation given by Klein[64]:

$$\theta_r = 89.8 - 0.5788\beta + 0.002693\beta^2 \qquad (3.64)$$

Monthly average transmittance-absorptance product for beam radiation $(\overline{\tau\alpha})_b$, is calculated from the Equation (3.61), by evaluating θ_i from:

$$\theta_i = \cos^{-1}\left[\cos(\phi_L-\beta)\cos\delta\cos\left(\frac{5\pi}{24}\right)+\sin(\phi_L-\beta)\sin\delta\right] \qquad (3.65)$$

3.7. YEARLY ENERGY BALANCES

Stored solar energy extracted from the storage and heat pump work are energy transferred to the heating system. A fraction of the stored energy is lost to the earth surrounding the energy storage, and remaining part of the stored energy is used to heat the house in heating season.

The annual dimensionless useful solar energy gain is obtained by integrating monthly components of useful solar energy given in Equation (3.39) over the year.

$$q_{sy} = \int_{Year} q_u(\tau)d\tau \tag{3.66}$$

The monthly dimensionless heat pump work is calculated from the Equation (3.35). The annual dimensionless heat pump work can be estimated by evaluating the following integral.

$$w_y = \int_{Year} w(\tau)^+ d\tau \tag{3.67}$$

The winter house heat requirement is estimated from the following integral:

$$q_{hy} = \int_{Year} q_h(\tau)^+ d\tau \tag{3.68}$$

The superscript + in equations (3.67)-(3.68) implies that only positive values of the integrand should be accounted for when estimating the integral. Annual net heat addition to the storage is estimated by integration of $q(\tau)$ given in Equation (3.23):

$$q_y = \int_{Year} q(\tau)\,d\tau \qquad (3.69)$$

Annual increase in the dimensionless internal energy of the storage is given by:

$$q_{st} = \int_{Year} p_s \frac{d\phi_w}{d\tau} d\tau \qquad (3.70)$$

or

$$q_{st} = \int_{Year} p_s\, d\phi_w \qquad (3.71)$$

Annual dimensionless heat loss to the earth surrounding the storage is given by:

$$q_{ly} = q_y - q_{st} \qquad (3.72)$$

Annual energy ratio is an important parameter for thermal performance of the heating system.

Annual solar energy ratio is the ratio of annual solar energy gain to the total annual energy consumption by the heating system.

$$R_S = \frac{q_{sy}}{q_{sy} + \frac{w_y}{\gamma}} \qquad (3.73)$$

Annual heat pump work ratio is the ratio of annual heat pump work to the energy consumed by the heating system.

$$R_W = \frac{w_y}{q_{sy} + \dfrac{w_y}{\gamma}} \tag{3.74}$$

The ratio of the annual energy loss to the energy consumption by the heating system can be expressed by:

$$R_L = \frac{q_{ly}}{q_{sy} + \dfrac{w_y}{\gamma}} \tag{3.75}$$

Annual house heat load ratio is the ratio of annual house heat load to the total energy consumption.

$$R_H = \frac{q_{hy}}{q_{sy} + \dfrac{w_y}{\gamma}} \tag{3.76}$$

An index of the performance of the heat pump system is the coefficient of performance of the heat pump. By definition, the actual coefficient of performance of a heat pump, during the heating cycle, is equal to the ratio of instantaneous heat output, q_{hy} at stated conditions, divided by the net work required, w_y to produce the effect. Annual coefficient of performance of heat pump is given by:

$$COP = \frac{q_{hy}}{\dfrac{w_y}{\gamma}} \tag{3.77}$$

The performance of a solar collector can be measured in terms of the collector efficiency which is defined as the ratio of the useful energy gain over some specific time interval to the incident solar radiation over the same interval. Monthly average collector efficiency can be calculated from the following expression.

$$\eta_c = \frac{A_c F_R(\overline{\tau\alpha})\overline{H}_T\overline{\phi}}{A_c\overline{H}_T} \tag{3.78}$$

or

$$\eta_c = F_R(\overline{\tau\alpha})\overline{\phi} \tag{3.79}$$

The annual solar fraction is another important indicator for performance of solar aided ground coupled heat pump heating systems. This factor is defined as the ratio between the annual energy supplied by the solar collector to the total energy required at the load and can be calculated from the expression.

$$f = \frac{q_{sy} - q_{ly}}{q_{hy}} \tag{3.80}$$

or

$$f = \frac{q_{hy} - \frac{w_y}{\gamma}}{q_{hy}} \tag{3.81}$$

3.8. DESIGN PARAMETERS USED IN CALCULATIONS

The present model for simulating the performance of a solar aided ground coupled heat pump system is used to study the effects of different parameters. A standard set of the parameters are defined and their effects on the system thermal performance are investigated. Some parameters are found to have a more dominant effect on the performance of the ground coupled heat pump heating system. These parameters are meteorological parameters, thermal and physical properties of the earth, collector parameters, ground reflectance, and storage system parameters. The parameters used in calculations are briefly discussed in detail in the following sections.

3.8.1. METEOROLOGICAL PARAMETERS

Temporal variation of temperature in the thermal energy storage and thermal performance of the heating system are functions of meteorological parameters. In the present study, these parameters for five Locations in Turkey are used in the calculations to investigate the influence of these parameters. The locations considered are Ankara, Elazığ, Gaziantep, İstanbul, and İzmir. The parameters consist of latitude angle ϕ_L, outside design air temperature T_d, monthly average outside air temperature $\overline{T}_a$, and monthly average solar radiation on a horizontal surface $\overline{H}$. The outside design air temperature and the other parameters are taken respectively from Dağsöz[65] and Ünsal and Doğantan[66]. Values of these parameters for the cities under consideration are given in Table 3.1.

Table 3.1. Meteorological parameters for five locations

Months	**Ankara** ϕ_L = 40.0 ° T_d = °C		**Elazığ** ϕ_L = 38.1 ° T_d = -12 °C		**Gaziantep** ϕ_L = 37.1 ° T_d = -9 °C		**İstanbul** ϕ_L = 40.0 ° T_d = -3 °C		**İzmir** ϕ_L = 38.5 ° T_d = 0 °C	
	$\overline{T}_a$ (°C)	$\overline{H}$ Mj/ m^2day	$\overline{T}_a$ (°C)	$\overline{H}$ Mj/ m^2day	$\overline{T}_a$ (°C)	$\overline{H}$ Mj/ m^2day	$\overline{T}_a$ (°C)	$\overline{H}$ Mj/ m^2day	$\overline{T}_a$ (°C)	$\overline{H}$ Mj/ m^2day
Jul	23.1	23.7	27.2	20.9	27.1	22.1	23.2	21.5	27.5	19.0
Aug	23.3	22.3	27.0	19.9	26.9	21.1	19.7	19.8	27.0	17.6
Sep	18.4	17.6	22.0	15.6	22.2	16.8	15.5	14.9	22.7	13.9
Oct	12.9	12.5	14.8	10.2	15.3	11.7	11.9	10.0	18.3	10.1
Nov	7.7	7.4	7.8	5.9	9.4	7.9	8.0	6.7	13.9	6.4
Dec	2.5	4.9	1.5	4.6	4.5	5.7	5.1	4.5	10.4	4.4
Jan	0.3	6.0	-1.3	4.9	2.6	6.2	5.5	4.4	8.2	5.1
Feb	1.0	8.5	0.0	8.0	3.6	9.2	6.7	7.4	8.8	7.1
Mar	4.7	12.7	4.7	10.9	7.2	12.6	10.9	11.6	11.1	10.3
Apr	11.1	17.9	11.8	15.3	12.7	17.7	15.8	16.2	15.3	14.4
May	16.1	20.9	17.4	17.2	18.2	18.3	20.6	18.9	20.1	19.1
Jun	20.0	22.1	22.9	21.1	23.7	22.6	23.2	21.3	24.7	18.3

3.8.2.THERMAL AND PHSICAL PROPERTIES OF EARTH

Thermal and physical properties of the geological structure are the parameters affecting the temperature of the storage and the thermal performance of the heating system. The thermal and physical parameters of coarse gravelled earth, sand, granite and limestone were taken from Özışık[67] and given in Table 3.2.

Table 3.2. Thermal and physical properties of the geological structure

Earth Type	Density (Kg/m^3)	Thermal Conductivity (W/m°C)	Thermal Diffusivity (m^2/sec)	Heat Capacity (J/kg°C)
Coarse gravelled	2050	0.519	$1.39x10^{-7}$	1842
Sand	1500	0.3	$2.50\,x10^{-7}$	800
Granite	2640	3.0	$1.40\,x10^{-7}$	820
Limestone	2500	1.3	$5.75\,x10^{-7}$	900

3.8.3. COLLECTOR PARAMETERS

Solar collectors are special kinds of heat exchangers which transfer the radiant energy of incident sunlight to the thermal energy of a working fluid, usually liquid or air. Flat plate collectors are commonly used in low and medium temperature applications. They are useful in supplying thermal energy at moderate temperatures, up to the normal boiling point of water. The major applications of flat plate collectors are in domestic hot water and space heating, and to a lesser degree, in industrial processes. For this reason, in the present study, the parameters corresponding to flat plate collectors are used in the calculations.

The solar collector construction, orientation, and operating temperature, combined with meteorological conditions: solar radiation level, air temperature, and wind speed and fluid flow rate through the collector are the primary factors affecting performance.

There is a number of variable parameters which are important in the design of a residential active solar heating system. Those related to the collector are the collector area and the collector slope. Collector parameters have an effect on the useful solar energy gain. The water temperature in the energy storage and the thermal performance of the heat pump system is a function of the useful energy gain. Four different collectors are used in the present calculations. These collectors are one glass cover black paint, two glass cover black paint, one glass cover selective surface and two glass cover selective surface type collectors. Parameters of these collector types as given by Simon[63] are tabulated in Table 3.3.

Table 3.3. Collector parameters used in calculations

Collector Type	**b_o**	**$(\tau\alpha)_n$**	α	ε	**U_L,W/m^{2}C**
Black paint-1 glass	0.078	0.89	0.97	0.97	7.4
Black paint-2 glass	0.15	0.76	0.97	0.97	4.5
Selective Surface-1 glass	0.11	0.80	0.90	0.1	5
Selective Surface-2 glass	0.16	0.74	0.95	0.07	3.2

In a solar system, collector orientation is also an important design consideration. Fixed position flat plate collectors are generally placed towards the south. In the present computer simulation, collectors are directed toward the south and the value of collector heat removal factor F_R, is taken as 0.75. Collector slope effects the storage temperature and thermal performance of the heat pump system. In the present calculations, collector slope is varied from 0 to 90 in steps of 10 to investigate influence of collector slope on the system performance.

3.8.4. MONTHLY AVERAGE GROUND REFLECTANCE

Available solar energy gain by a solar collector depends upon the amount of radiation on a tilted surface. A tilted surface also receives solar radiation reflected from the ground and other surroundings. The ground and other surfaces seen by the tilted surface are assumed to have a diffuse reflectivity of ρ_g. Lui and Jordan[57] suggest that monthly average ground reflectivity, ρ_g varies from 0.2 to 0.7 depending upon the extent of snow cover. The monthly average ground reflectivity, ρ_g used in the present calculations is give in Table 3.4.

Table 3.4. Monthly average ground reflectance

Month	July	Aug	Sep	Oct	Nov	Dec	Jan	Feb	Mar	Apr	May	Jun
ρ_g	0.2	0.2	0.2	0.2	0.2	0.2	0.3	0.7	0.5	0.3	0.2	0.2

3.8.5. STORAGE PARAMETERS

The water temperature in energy storage changes with the volume of the storage. The volume of the energy storage buried in deep earth is an important variable which effects the system performance. Five different storage volumes are selected corresponding to the radii of 3, 4, 5, 6, and 7 meters for one house. Thermal performance for large heating systems is investigated considering one hundred times the single house storage volume and considering heating loads corresponding to a heating load of 100 houses.

CHAPTER IV

MODELING OF A SOLAR AIDED HEAT PUMP SPACE HEATING SYSTEM WITH A SEASONAL HEMISPHERICAL UNDERGROUND THERMAL ENERGY STORAGE

4.1. INTRODUCTION

In this chapter, annual periodic performance of a solar assisted ground-coupled heat pump system with a seasonal hemispherical underground thermal energy storage is investigated using analytical and computational methods. The geometry of the ground-coupled heat pump system is depicted in Figure 4.1. The heat pump system as shown in Figure 4.1 consists of solar collectors, a heat pump, a house to be heated, and a seasonal underground hemispherical storage. Solar collectors mounted on the roof of the house are operated over the whole year charging energy to the thermal energy storage. The amount of charged energy for each month is estimated using the utilizability, $\bar{\phi}$, method. Formulation of the $\bar{\phi}$ method is given in Chapter 3. The charged energy to the storage is extracted by the heat pump and it is supplied to the house. Modelling of the heat pump, calculation of the house heat load, and operating conditions of the ground-coupled heat pump system are the same as those explained in Chapter 3 and will not be repeated in this chapter.

Seasonal underground thermal energy storage is a part of the solar assisted ground-coupled heat pump system. The storage plays an important role in this system. It is necessary to solve for transient temperature distribution in earth outside the storage for problem integration of the storage with other components of the heating system for proper completion of the model and for investigating the annual performance of the heating system. Therefore, an analytical solution procedure for a transient temperature field outside a buried hemispherical underground solar thermal energy storage is presented in the next section. The problem for the transient temperature outside the seasonal thermal energy storage is formulated first. The formulation is then put into dimensionless form using dimensionless variables. The transient heat transfer problem is then analysed by applications of the Complex Finite Fourier (CFFT) and Finite Bessel Transforms (FFT). A solution procedure for eigenvalue problem arising in from the transient heat transfer problem is presented. The solution for the temperature field problem is then coupled with the formulations for the other components of the system as explained in Chapter 3. The model is then used to determine the annual variation of the temperature of water in the thermal energy storage and to investigate the long term performance of a the heat pump system with a seasonal hemispherical energy storage. An interactive computer program based on the analytical model of the system was prepared in Fortran77 for computing the transient temperature field in earth, temperature of water in the storage and the annual periodic performance of the heat pump system described.

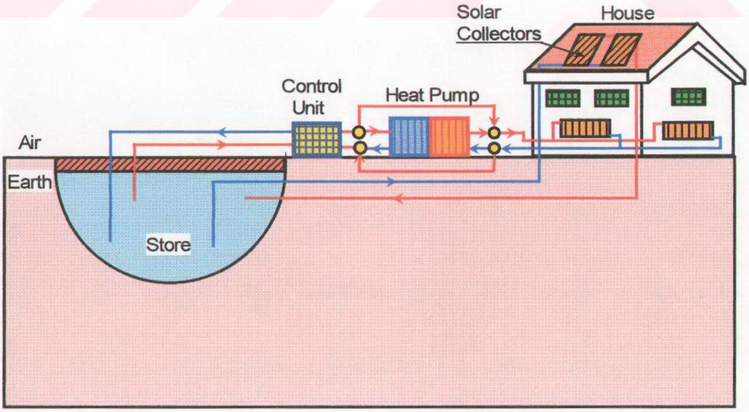


Figure 4.1 Heat pump system with hemispherical storage in earth.

4.2. ANALYSIS OF THE TRANSIENT HEAT TRANSFER PROBLEM FOR THE HEMİSPHERİCAL STORAGE

Schematic representation of the transient heat transfer problem considered is shown in Figure 4.2. The water filled hemispherical thermal energy storage is assumed to be located below the ground level. Top surface of the storage tank is covered with an insulating material. Far-field temperature away from the storage is taken equal to the deep ground temperature T_∞ which is assumed to be constant with respect to time. The water in the storage is assumed to be at a fully mixed and spatially lumped time varying temperature $T_w(t)$. Earth outside the storage is assumed to have homogeneous structure, constant thermal and physical properties everywhere. The temperature of the earth surrounding the storage $T = T(r,\theta ,t)$ is a function of the radial coordinate measured from the centre of the spherical storage, angle measured from the bottom of the storage shown in Figure 4.2 and time, t. The temperature in earth is assumed to be symmetrical about the polar angle.

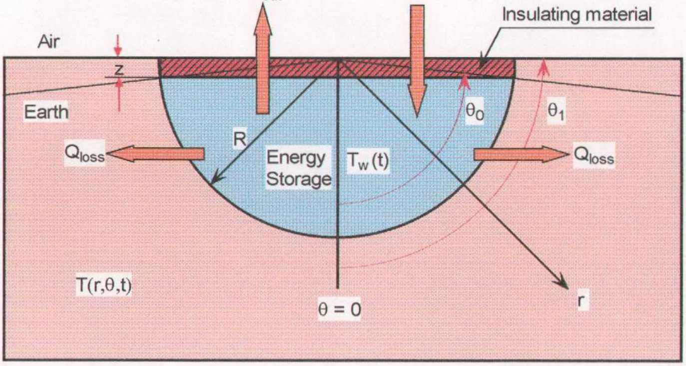


Figure 4.2. Schematic representation of system under study

The problem formulation for annually periodic transient temperature field outside the hemispherical thermal energy storage is given by the following partial differential equation, and initial and boundary conditions all in the spherical coordinate system.

$$\frac{\partial}{\partial r}\left[r^2\frac{\partial T}{\partial r}\right]+\frac{1}{\sin\theta}\frac{\partial}{\partial\theta}\left[\sin\theta\frac{\partial T}{\partial\theta}\right]=\frac{r^2}{\alpha}\frac{\partial T}{\partial t} \tag{4.1}$$

$$T(R,\theta,t)=T_w(t) \tag{4.2}$$

$$T(R_\infty,\theta,t)=T_\infty \tag{4.3}$$

$$\frac{\partial T}{\partial\theta}(r,0,t)=0 \tag{4.4}$$

$$-\frac{\partial T}{\partial\theta}(r,\theta_0,t)=B_i\left[T(r,\theta_0,t)-T_a(t)\right] \tag{4.5}$$

Where, Bi is the Biot number and defined as:

$$Bi=\frac{U'}{k}=\frac{U\,r}{k} \tag{4.6}$$

α, given in equation (4.1), is the thermal diffusivity of the earth outside the storage. The energy balance relating energy charge rate to the storage, energy accumulation rate in the storage, the energy loss rate to the outside air from top surface of the storage and the conduction heat loss rate to the surrounding earth is given as follows:

$$Q=\rho_w c_w V_w\frac{dT_w}{dt}+U_T A_T\left[T_w(t)-T_a(t)\right]-k\int_0^{2\pi}\int_0^{\theta_0}R^2\sin\theta d\phi\frac{\partial T}{\partial r}(R,\theta,t)d\theta \tag{4.7}$$

where q_k, ϕ_k and ϕ_{ak} are complex Fourier coefficients of dimensionless net heat input rate to the storage, water temperature in the storage and outside air temperature respectively. The transformed formulation of the problem given by the Equations (4.17)-(4.22) will be made homogeneous in the θ-direction for rendering the problem solvable by application of a finite Hankel transform. The following is introduced.

$$\psi_k = x(\phi_k - \phi_{ak}) \tag{4.23}$$

When the transformation given by equation (4.23) is introduced into the problem defined by equation (4.17) through (4.22), the following problem is obtained.

$$x^2 \frac{\partial^2 \psi_k}{\partial x^2} + \frac{1}{\sin\theta} \frac{\partial}{\partial \theta}\left[\sin\theta \frac{\partial \psi_k}{\partial \theta}\right] = x^2 i v_k \psi_k + x^3 i v_k \phi_{ak} \tag{4.24}$$

$$\psi_k(1,\theta,v_k) = \phi_{wk} - \phi_{ak} \tag{4.25}$$

$$\psi_k(x_\infty,\theta,v_k) = -x_\infty \phi_{ak} \tag{4.26}$$

$$\frac{\partial \psi_k}{\partial \theta}(x,0,v_k) = 0 \tag{4.27}$$

$$\frac{\partial \psi_k}{\partial \theta}(x,\theta_0,v_k) + Bi_\theta \psi_k(x,\theta_0,v_k) = 0 \tag{4.28}$$

$$q_k = p i v_k \phi_{wk} + s(\phi_{wk} - \phi_{ak}) - \int_0^{\theta_0} \sin\theta \left\{\psi_k(1,\theta,v_k) - \frac{\partial \psi_k}{\partial x}(1,\theta,v_k)\right\} d\theta \tag{4.29}$$

The problem posed by equations (4.24)-(4.28) will be decomposed into two problems for solution by superposition principle. For this purpose, the following transformation is introduced:

$$\psi_k(x,\theta,\nu_k) = \psi_{1k}(x,\theta,\nu_k) + \psi_{2k}(x,\nu_k) \tag{4.30}$$

4.2.2. Solution for ψ_{1k}

ψ_{1k} is forced to satisfy:

$$x^2 \frac{\partial^2 \psi_{1k}}{\partial x^2} + \frac{1}{\sin\theta}\frac{\partial}{\partial\theta}\left[\sin\theta \frac{\partial \psi_{1k}}{\partial\theta}\right] = x^2 i\nu_k \psi_{1k} \tag{4.31}$$

$$\psi_{1k}(1,\theta,\nu_k) = \phi_{wk} - \phi_{ak} \tag{4.32}$$

$$\psi_{1k}(x_\infty,\theta,\nu_k) = 0 \tag{4.33}$$

$$\frac{\partial \psi_{1k}}{\partial\theta}(x,0,\nu_k) = 0 \tag{4.34}$$

$$\frac{\partial \psi_{1k}}{\partial\theta}(x,\theta_0,\nu_k) + Bi_\theta \psi_{1k}(x,\theta_0,\nu_k) = 0 \tag{4.35}$$

The problem expressed by (4.31)-(4.35) can be solved by application of the following integral Hankel transform.

$$\psi_{1kn}(x,\lambda_n,\nu_k) = \int_0^{\theta_0} \psi_{1k}(x,\theta,\nu_k)\sin\theta\, K_n(\theta)\, d\theta \tag{4.36}$$

$$\psi_{1k}(x,\theta,\nu_k) = \sum_{n=0}^{\infty} \psi_{1kn}(x,\lambda_n,\nu_k) K_n(\theta) \tag{4.37}$$

A kernel for this integral transform will be obtained from solution of the following eigenvalue problem:

$$\frac{1}{\sin\theta}\frac{d}{d\theta}\left[\sin\theta\frac{dK}{d\theta}\right]+\lambda^2K=0 \tag{4.38}$$

$$\frac{dK}{d\theta}(0)=0 \tag{4.39}$$

$$\frac{dK}{d\theta}(\theta_0)+B_{i\theta}K(\theta_0)=0 \tag{4.40}$$

Solution procedure of the eigenvalue problem defined (4.38)-(4.40) is explained in section (4.2.5) in detail.

When the integral Hankel transform given by (4.36) is applied to the problem for ψ_{1k} defined by (4.31)-(4.35), the following problem consisting of a differential equation and boundary conditions is obtained.

$$\frac{d^2\psi_{1kn}}{dx^2}-\left(i\nu_k+\frac{\lambda_n{}^2}{x^2}\right)\psi_{1kn}=0 \tag{4.41}$$

$$\psi_{1kn}(1,\lambda_n,\nu_k)=\beta_n(\phi_{wk}-\phi_{ak}) \tag{4.42}$$

$$\psi_{1kn}(x_\infty,\theta,\nu_k)=0 \tag{4.43}$$

Where

$$\beta_n(\theta)=\int_0^{\theta_0}\sin\theta K_n(\theta)d\theta \tag{4.44}$$

The solution of the problem consisting of Equations (4.41)-(4.43) has three different forms depending on the transformation indice k. The solutions for k<0, k=0, and k>0 are given as:

$$\psi_{1kn}(x,\lambda_n,\nu_k)=\beta_n(\phi_{wk}-\phi_{ak})\,x^{1/2}\,\frac{H_{\mu_n}^{(1)}(\omega_k x)}{H_{\mu_n}^{(1)}(\omega_k)} \qquad \text{For } k<0 \qquad (4.45)$$

$$\psi_{10n}(x,\lambda_n,\nu_k)=\beta_n(\phi_{w0}-\phi_{a0})\,x^{\left(\frac{1}{2}-\mu_n\right)} \qquad \text{For } k=0 \qquad (4.46)$$

$$\psi_{1kn}(x,\lambda_n,\nu_k)=\beta_n(\phi_{wk}-\phi_{ak})\,x^{1/2}\,\frac{H_{\mu_n}^{(2)}(\omega_k x)}{H_{\mu_n}^{(2)}(\omega_k)} \qquad \text{For } k>0 \qquad (4.47)$$

Where, $H_{\mu_n}^{(1)}(\omega_k x)$ and $H_{\mu_n}^{(2)}(\omega_k x)$ are Hankel functions of the first and second kinds of real order, μ_n, and with complex arguments. These functions are defined as:

$$H_{\mu_n}^{(1)}(\omega_k x)=J_{\mu_n}(\omega_k x)+iY_{\mu_n}(\omega_k x) \qquad (4.48)$$

$$H_{\mu_n}^{(1)}(\omega_k)=J_{\mu_n}(\omega_k)+iY_{\mu_n}(\omega_k) \qquad (4.49)$$

$$H_{\mu_n}^{(2)}(\omega_k x)=J_{\mu_n}(\omega_k x)-iY_{\mu_n}(\omega_k x) \qquad (4.50)$$

$$H_{\mu_n}^{(2)}(\omega_k)=J_{\mu_n}(\omega_k)-iY_{\mu_n}(\omega_k) \qquad (4.51)$$

Where

$$\mu_n = \sqrt{\frac{1}{4}+\lambda_n^2} \tag{4.52}$$

and

$$\omega_k = \sqrt{-i\nu_k} \tag{4.53}$$

When the solutions given by (4.45)-(4.47) are substituted into the inverse Finite Hankel Transform Equation (4.37). The following solution ψ_{1k} defined as Equations (4.31)-(4.35) is obtained.

$$\psi_{1k}(x,\theta,\nu_k) = \sum_{n=0}^{\infty} \beta_n K_n(\theta) \left((\phi_{wk} - \phi_{ak}) x^{1/2} \frac{H^{(1)}_{\mu_n}(\omega_k x)}{H^{(1)}_{\mu_n}(\omega_k)} \right)$$

$$+ \sum_{n=0}^{\infty} \beta_n K_n(\theta) \left((\phi_{w0} - \phi_{a0})\, x^{\left(\frac{1}{2}-\mu_n\right)} \right) \tag{4.54}$$

$$+ \sum_{n=0}^{\infty} \beta_n K_n(\theta) \left((\phi_{wk} - \phi_{ak})\, x^{1/2} \frac{H^{(2)}_{\mu_n}(\omega_k x)}{H^{(2)}_{\mu_n}(\omega_k)} \right)$$

4.2.3. Solution for ψ_{2k}

ψ_{2k} is forced to satisfy:

$$\frac{d^2\psi_{2k}}{\partial x^2} - i\nu_k \psi_{2k} = x i \nu_k \phi_{ak} \tag{4.55}$$

$$\psi_{2k}(1,\nu_k) = 0 \tag{4.56}$$

$$\psi_{2k}(x_\infty,\nu_k) = -x_\infty \phi_{ak} \tag{4.57}$$

When the non-homogeneous second order differential equation given in (4.55) is solved subjected to the boundary conditions given by Equations (4.56)-(4.57), the solution for ψ_{2k} is obtained as:

$$\psi_{2k}(x, v_k) = \phi_{ak}\left(e^{(1-x)\sqrt{iv_k}} - x\right) \tag{4.58}$$

4.2.4. General solution of the problem

General solution of the problem defined as (4.24)-(4.28) is obtained by substituting the solution of the decomposed problem consisting of ψ_{1k} and ψ_{2k} into the Equation (4.30):

$$\begin{aligned}\psi_k(x, \theta, v_k) = &\sum_{n=0}^{\infty} \beta_n K_n(\theta)\left((\phi_{wk} - \phi_{ak})x^{1/2}\frac{H_{\mu_n}^{(1)}(\omega_k x)}{H_{\mu_n}^{(1)}(\omega_k)}\right) + \phi_{ak}\left(e^{(1-x)\sqrt{iv_k}} - x\right)\\ &+ \sum_{n=0}^{\infty} \beta_n K_n(\theta)\left[(\phi_{w0} - \phi_{a0})x^{\left(\frac{1}{2} - \mu_n\right)}\right] + \phi_{a0}(1-x)\\ &+ \sum_{n=0}^{\infty} \beta_n K_n(\theta)\left((\phi_{wk} - \phi_{ak})x^{1/2}\frac{H_{\mu_n}^{(2)}(\omega_k x)}{H_{\mu_n}^{(2)}(\omega_k)}\right) + \phi_{ak}\left(e^{(1-x)\sqrt{iv_k}} - x\right)\end{aligned} \tag{4.59}$$

When the solution for ψ_k given by Equation (4.59) is inserted into the inverse CFFT transform, for ψ_k the general solution for ψ is obtained as:

$$\psi(x,\theta,\tau) = \sum_{k=-\infty}^{-1} \left[\sum_{n=0}^{\infty} \beta_n K_n(\theta) \left((\phi_{wk} - \phi_{ak}) x^{1/2} \frac{H^{(1)}_{\mu_n}(\omega_k x)}{H^{(1)}_{\mu_n}(\omega_k)} \right) + \phi_{ak} \left(e^{(1-x)\sqrt{i\nu_k}} - x \right) \right] e^{i\nu_k \tau}$$

$$+ \sum_{n=0}^{\infty} \beta_n K_n(\theta) \left[(\phi_{w0} - \phi_{a0}) x^{\left(\frac{1}{2} - \mu_n\right)} \right] + \phi_{a0}(1-x)$$

$$+ \sum_{k=1}^{\infty} \left[\sum_{n=0}^{\infty} \beta_n K_n(\theta) \left((\phi_{wk} - \phi_{ak}) x^{1/2} \frac{H^{(2)}_{\mu_n}(\omega_k x)}{H^{(2)}_{\mu_n}(\omega_k)} \right) + \phi_{ak} \left(e^{(1-x)\sqrt{i\nu_k}} - x \right) \right] e^{i\nu_k \tau}$$

(4.60)

General solution for the transient temperature in the earth outside the thermal energy storage given the Equations (4.9)-(4.13) is obtained by using Equation (4.60) in Equation (4.23).

$$\phi(x,\theta,\tau) = X^{-1}\, \psi(x,\theta,\tau) + \phi_a(\tau) \tag{4.61}$$

The general solution of the transient temperature in earth is a function of complex Fourier coefficients of net heat input to the storage q_k, ambient air temperature, ϕ_{ak}, and water temperature ϕ_{wk} in the storage. In order to calculate the annual temperature distribution of the earth and water in the storage, q_k, ϕ_{ak}, and ϕ_{wk} must be estimated. Estimation of q_k was given in Chapter 3. Estimation procedure of the ϕ_{wk}, and ϕ_{ak} is explained in the following sections of (4.2.5) and (4.2.6)

4.2.5. Estimation of the Complex Fourier Coefficient for Water Temperature

The complex Fourier coefficients of water temperature ϕ_{wk} in side the storage is obtained using the solution of the decomposed problem for ψ_{1k}

and ψ_{2k} in the transformed energy balance Equation (4.29) for three different cases as follows:

$$\phi_{wk} = \frac{q_k + \left[s + \sum_{n=0}^{\infty} \beta_n^2 \Theta_1(\lambda_n, \nu_k) + \left(\cos\theta_0 - 1\right)\left(1 + \sqrt{i\nu_k}\right) \right] \phi_{ak}}{pi\nu_k + s + \sum_{n=0}^{\infty} \beta_n^2 \Theta_1(\lambda_n, \nu_k)} \quad \text{For } k < 0 \quad (4.62)$$

$$\phi_{w0} = \frac{q_0 + \left[s + \sum_{n=0}^{\infty} \beta_n^2 \left(\frac{1 + 2\mu_n}{2} \right) + \left(\cos\theta_0 - 1\right) \right] \phi_{a0}}{s + \sum_{n=0}^{\infty} \beta_n^2 \left(\frac{1 + 2\mu_n}{2} \right)} \qquad \text{For } k = 0 \quad (4.63)$$

$$\phi_{wk} = \frac{q_k + \left[s + \sum_{n=0}^{\infty} \beta_n^2 \Theta_2\left(\lambda_n, \nu_k\right) + \left(\cos\theta_0 - 1\right)\left(1 + \sqrt{i\nu_k}\right) \right] \phi_{ak}}{pi\nu_k + s + \sum_{n=0}^{\infty} \beta_n^2 \Theta_2\left(\lambda_n, \nu_k\right)} \quad \text{For } k > 0 \quad (4.64)$$

where

$$\Theta_1(\lambda_n, \nu_k) = 1 - \left(\frac{1 + 2\mu_n}{2} - \omega_k \frac{H^{(1)}_{\mu_{n+1}}(\omega_k)}{H^{(1)}_{\mu_n}(\omega_k)} \right) \qquad (4.65)$$

$$\Theta_2(\lambda_n, \nu_k) = 1 - \left(\frac{1 + 2\mu_n}{2} - \omega_k \frac{H^{(2)}_{\mu_{n+1}}(\omega_k)}{H^{(2)}_{\mu_n}(\omega_k)} \right) \qquad (4.66)$$

and

$$H^{(1)}_{\mu_{n+1}}(\omega_k) = J_{\mu_{n+1}}(\omega_k) + iY_{\mu_{n+1}}(\omega_k) \tag{4.67}$$

$$H^{(2)}_{\mu_{n+1}}(\omega_k) = J_{\mu_{n+1}}(\omega_k) - iY_{\mu_{n+1}}(\omega_k) \tag{4.68}$$

4.2.6. A Fourier Series Representation for Ambient Air Temperature

The annual periodic temperature inside earth given by Equation (4.60) depends on the complex Fourier coefficients of the dimensionless ambient air temperature, $\phi_a(\tau)$. The dimensionless temperature, $\phi_a(\tau)$ may be expressed by a Fourier series with a period of one year. Defining ϕ_{ai} to be monthly components of the dimensionless ambient air temperature, the dimensionless temperature, $\phi_a(\tau)$ may be expressed by the following vector.

$$\phi_a(\tau) = (\phi_{a1}, \phi_{a2}, \phi_{a3}, \phi_{a4}, \phi_{a5}, \phi_{a6}, \phi_{a7}, \phi_{a8}, \phi_{a9}, \phi_{a10}, \phi_{a11}, \phi_{a12}) \tag{4.69}$$

The dimensionless ambient air temperature, $\phi_a(\tau)$ given by Equation (4.69) can be expressed by the following Fourier series with a period of one year.

$$\phi_a(\tau) = \sum_{n=-\infty}^{+\infty} \phi_{ak}\, e^{i\nu_k \tau} \tag{4.70}$$

Complex Fourier coefficients of the dimensionless ambient air temperature, ϕ_{ak}, in Equation (4.70) are given as:

$$\phi_{a0} = \frac{1}{12}\sum_{j=1}^{12}\phi_{aj} \qquad \text{for } k = 0 \qquad (4.71)$$

$$\phi_{ak} = \frac{1}{2\pi k}\sum_{j=1}^{12}\phi_{aj}\left\{\eta_{3,j} + i\eta_{4,j}\right\} \qquad \text{fork} \ge 1 \qquad (4.72)$$

where

$$\eta_{3,j} = \sin(2\pi k r_j) - \sin(2\pi k r_{j-1}) \qquad (4.73)$$

$$\eta_{4,j} = \cos(2\pi k r_j) - \cos(2\pi k r_{j-1}) \qquad (4.74)$$

4.2.7. Solution of The Eigenvalue Problem

In this section, a solution procedure for the eigenvalue problem in θ-direction defined by Equations (4.38)-(4.40) will be given. The eigenvalue problem may be expressed in the following form.

$$\frac{d^2K}{d\theta^2} + \frac{1}{\theta}\frac{dK}{d\theta} + \varepsilon\left[\cot\theta - \frac{1}{\theta}\right]\frac{dK}{d\theta} + \lambda^2 K = 0 \qquad (4.75)$$

$$\frac{dK}{d\theta}(0) = 0 \qquad (4.76)$$

$$\frac{dK}{d\theta}(\theta_0) + B_{i\theta}K(\theta_0) = 0 \qquad (4.77)$$

Where, ε is equal to unity. A solution procedure this eigenvalue problem is explained below. Let us consider the following simple eigenvalue problem by equating the ε is zero.

$$\frac{d^2\Phi}{d\theta^2}+\frac{1}{\theta}\frac{d\Phi}{d\theta}+\alpha^2\Phi=0 \tag{4.78}$$

$$\frac{d\Phi}{d\theta}(0)=0 \tag{4.79}$$

$$\frac{d\Phi}{d\theta}(\theta_0)+B_{i\theta}\Phi(\theta_0)=0 \tag{4.80}$$

Normalized eigenfunctions of this problem is given in Kakaç and Yener[68] as:

$$\Phi_n(\theta,\alpha_n)=\frac{\sqrt{2}\alpha_n}{\theta_0\left(B_{i\theta}^2+\alpha_n^2\right)^{1/2}}\frac{J_0(\alpha_n\theta)}{J_0(\alpha_n\theta_0)} \tag{4.81}$$

When the n is equal to zero, $\alpha_0=0$, then, the corresponding normalised eigenfunction for this special case is:

$$\Phi_0(\theta,\alpha_0)=\frac{\sqrt{2}}{\theta_0} \tag{4.82}$$

Eigenvalues are positive roots of the following equation.

$$B_{i\theta}\, J_0(\alpha_n\theta_0)-\alpha_n^2\, J_1(\alpha_n\theta_0)=0 \tag{4.83}$$

Where Φ_n, α_n are eigenfunctions and eigenvalues of the problem defined by (4.78)-(4.80). Solution of the eigenvalue problem defined by Equations (4.75)-(4.77) can be written as.

$$K(\theta,\lambda)=\sum_{n=1}^{\infty} a_n \Phi_n(\theta,\alpha_n) \tag{4.84}$$

Where a_n's are constants and can be determined by inserting the expansion given by (4.84) into the differential equation of the eigenvalue problem defined by (4.75)-(4.77) multiplying by the weight function Φ_m and integrating over the problem domain. The final result is given by:

$$\sum_{n=1}^{\infty} a_n b_{nm} + \sum_{n=1}^{\infty} a_n (\lambda^2 - \alpha_n^2) c_{nm} = 0 \tag{4.85}$$

where

$$b_{nm} = \int_0^{\theta_0} (\theta \cos\theta - 1) \Phi_n' \, \Phi_m \, d\theta \tag{4.86}$$

and

$$c_{nm} = \int_0^{\theta_0} \theta \, \Phi_n \, \Phi_m \, d\theta = \begin{cases} 0 & \text{if } n \neq m \\ 1 & \text{if } n = m \end{cases} \tag{4.87}$$

a_n's were calculated by using matrix methods.

CHAPTER V

RESULTS AND DISCUSSION

5.1. INTRODUCTION

A computer simulation program in FORTRAN 77 based on two analytical models presented in Chapter 3 and 4 was developed for the numerical calculations. Computation procedure is explained in the following section in detail. The program was used to determine the transient temperature of the storage, transient temperature distribution inside the earth, as well as the thermal performance of the solar aided ground coupled heat pump system with seasonal spherical and hemispherical thermal storages.

The simulation program developed is executed using Lahey F77L-EM/32 compiler for varying values of the system input parameters. The results obtained from the execution are depicted in Figures and a discussion is given in this chapter. This chapter consists of four main sections which are: computation procedure for annually periodic operation, results obtained from the heating system with spherical storage, results obtained from the heating system with hemispherical storage, and comparison of the predicted results with the results from similar studies available in the literature.

5.2. COMPUTATION PROCEDURE FOR ANNUALLY PERIODIC OPERATION

Computational modeling of the solar assisted ground coupled heat pump system with seasonal thermal energy storage was utilized to develop a FORTRAN 77 simulation program called SOLARSTORE. The simulation program was then used to predict the long term annually periodic performance of the heat pump heating systems. The SOLARSTORE consists of a main code and five main subroutine subprograms. These are called COLRAD, TAGALFA, SIPSTORE, HEMISTORE and RESULT, all of which run on an WINDOWS 95 compatible microcomputer. The SOLARSTORE comprises some 8000 functional lines and requires 300 Kbytes disk space on the computer. A simplified outline of SOLARSTORE is shown in Figure 5.1. The detailed flowchart and listing of the program are given in Appendix B.

Main code takes input data which was divided into five distinctive groups: the weather data, the collector data, the earth data, the storage data, and the heat load side parametric data. The weather data for monthly average solar radiation incident on a horizontal surface and for monthly average outside air temperature together with the outside design air temperature in five cities were taken from Ünsal and Doğantan[66]. The cities considered are Ankara, Elazığ, Gaziantep, İstanbul, and İzmir. Parameters data for four types of flat collectors used in calculations is taken from Simon[63]. The solar collector data and the load side parametric data are given as data statements in the main code of SOLARSTORE. The earth data is for the thermal properties of four types of earth used in the present calculations. The values of thermal properties of earth used in this study are given in Table 3.2 which are given input via data statements in SOLARSTORE. The storage data, types of collectors, number of houses and types of earth outside the storage are provided interactively on monitor by the user.

The subprogram COLRAD computes monthly average of daily radiation on a tilted collector surface. It is also necessary to calculate the transmittance absorbtance product to obtain useful solar energy gain by the collectors. Monthly average transmittance absorbtance product for the collector was computed using the subprogram TAGALFA.

An iterative procedure was used to calculate water temperature in the storage and performance parameters for the space heating system with a spherical thermal storage using subprogram SIPSTORE. SIPSTORE performs among others, the following calculations: house heat load, heat pump work, monthly average daily useful energy collection, complex Fourier coefficients of the net heat input rate to the storage, storage and earth temperatures, and all of the performance parameters.

Energy requirements of the house during the heating season was estimated for each month separately using the degree-day method. The UA-value of one house was taken to be 345 Watts/°C which is the ratio of the design heat load of a house to the winter design inside/outside air temperature difference. The ratio of UA-value of the house to UA-value of load side heat exchanger, u, was taken as equal to 1.2. The winter design heat load of a single house considered in this study was taken 10 kW. Design house indoor temperature was taken 20 degrees Celsius throughout the heating season. An initial storage temperature equal to deep ground temperature was assumed for 12 months which was taken equal to 15 C. Monthly average value of the required daily heat pump work was then calculated from Equation (3.35) using the monthly average outside air temperatures, $\bar{T}_a$. Heat pump was operated only at times when the dimensionless storage temperature, ϕ_w, was less than the critical dimensionless temperature of $u(\phi_i - \phi_a) + \phi_i$. $\bar{\phi}$ was computed then for each month from Equation (3.47) using the initially assumed monthly values of the storage temperature. Monthly average daily useful solar energy gains, Q_u were calculated from Equation (3.39). Net energy charged to the storage was computed next from Equation (3.21)using the monthly house heat loads, monthly heat pump work values and monthly solar useful energy gains. Complex Fourier coefficients of the dimensionless energy charge rate to the storage were computed from Equations (3.26)-(3.27) using the CFCQ sub subprogram. By using these coefficients, a new set of monthly storage temperatures was calculated from Equation (3.12). Initially assumed storage temperatures and newly computed temperatures were then compared for each month. Iterative procedures was terminated when the difference between assumed and calculated storage temperatures were less than 0.01 °C for all 12 months. A new set of monthly storage temperatures for starting an iteration was calculated from $T = T_{old} + \alpha_R (T_{new} - T_{old})$ where α_R is a relaxation parameter. A value of α_R = 0.01 was sufficient for convergence. The present Fortran code calculates and reports the annually periodic thermal performance of the system. Finally the subprogram RESULT writes all outputs into a data file referred by the user.

The HEMISTORE subprogram was used to calculate temperature of the hemispherical storage and earth outside the storage based on the assumption of annually periodic operation of the system with the hemispherical storage. Eigenvalues λ_n and constants β_n were obtained from Equations (4.85) and (4.44) using a program named EIGEN. The program EIGEN is written in the Mathematica computer language, and its listing is given in Appendix A. The eigenvalues and constants obtained are utilized in data statements of HEMISTORE. Monthly storage temperatures, house heat loads, monthly heat pump work, monthly average daily useful solar energy gain, complex Fourier coefficients of dimensionless heat input rate of the storage temperature are calculated using the same procedure given in subprogram SIPSTORE. Complex Fourier coefficients of the dimensionless outside air temperature were then calculated from Equations (4.71)-(4.72) using subprogram CFCQ. Hankel functions of first and second kinds of real orders μ_n and μ_{n+1} given in Equations (4.48)-(4.51) were next computed using the subprogram CBESH. CBESH was downloaded from Guide to Available Mathematical Software (GAMS) on Internet (math.nist.gov). Listing of the CBESH is given in Appendix B. By using all these coefficients, a new set of storage and earth temperatures were calculated from Equation (4.61). Converged values of the storage and earth temperatures were obtained by the iteration procedure given for the subprogram SIPSTORE. HEMISTORE calculates annually periodic thermal performance of the space heating system using the converged storage and earth temperatures. At the end of this calculations, subprogram RESULT was used to write all output values to a data file referred by the user.

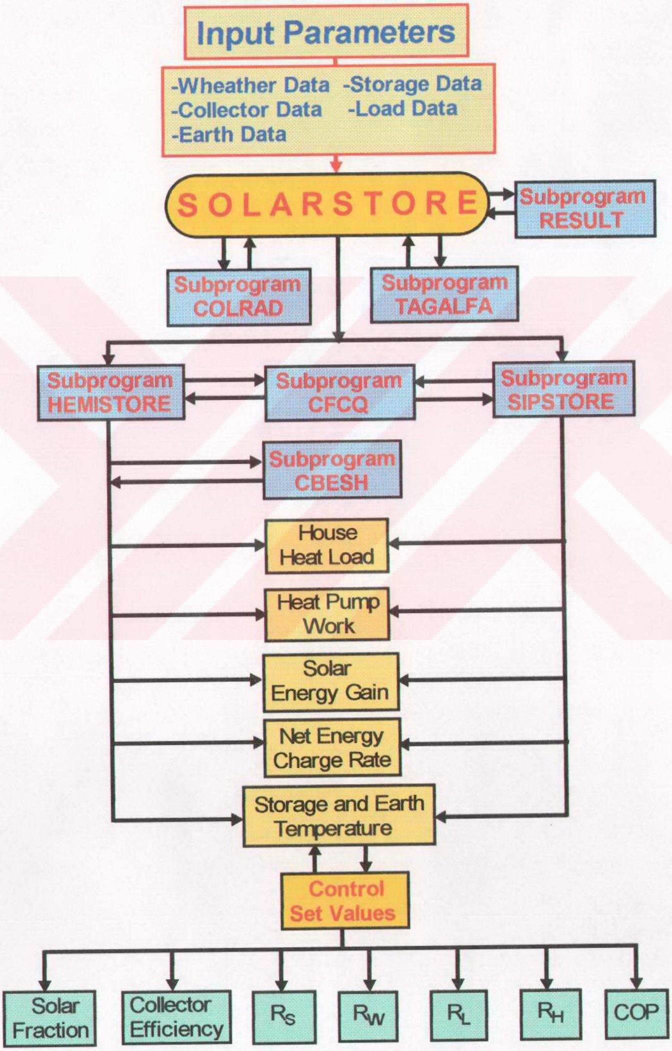


Figure 5.1. A simplified structure of the SOLARSTORE

5.3. RESULTS FOR HEATING SYSTEMS WITH SPHERICAL STORAGES

The simulation program, SOLARSTORE developed was executed to determine performances of a system with a load consisting of one house and of another system with a load consisting of 100 houses. Results obtained from the execution of the simulation program for space heating systems with seasonal underground spherical storages are presented in graphical forms and discussed in this section.

Figure 5.2 shows the effect of collector types on the water temperature in the storage embedded in limestone for one house. It is observed that the lowest storage temperature takes place when the collector type is selected for black paint-two glass cover. But the other collectors give approximately the same temperature for a month. So, the collector parameters corresponding to a black paint-1 glass cover collector, which is widely used in applications, was used in the discussions presented in this Chapter.

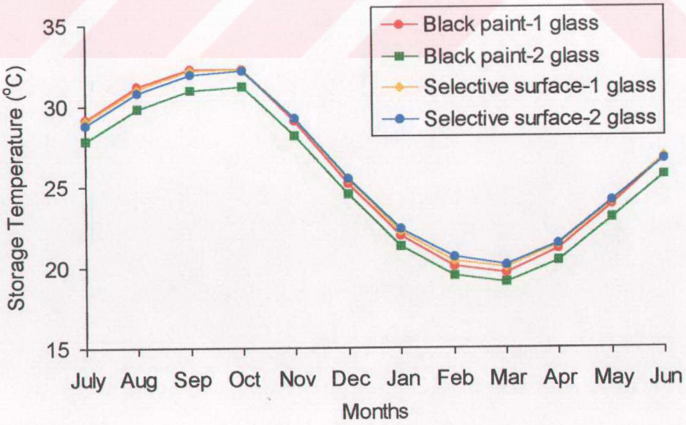


Figure 5.2. Effect of collector type on the water temperature in the storage embedded in limestone for one house (Gaziantep, $A_c = 30\ m^2$, a = 5 m)

Variation of monthly average solar radiation on a tilted surface with respect to collector slope is depicted in Figure 5.3 for Gaziantep. It is seen from the figure that the lowest annual radiation is obtained when collector slope is equal to 90. The highest annual radiation is obtained when collector slope is approximately equal to the latitude angle of the location. It was mentioned by Braun et. al.[69] that the optimum collector slope for space heating systems with seasonal storage is equal to the latitude of the particular location. So, unless specified otherwise, collector slope was taken equal to the latitude angle of all locations in this Chapter.

Thermal performance of a space heating system depends on temperature of water in the storage. Annual periodic variation of the thermal storage temperature versus collector slope is depicted in Figure 5.4 for a storage buried in limestone and for a system with one house. It is seen from this Figure that largest storage temperature is obtained when the collector slope is equal to latitude angle of Gaziantep. Smallest temperature occurs for the collector slope of 90 °.

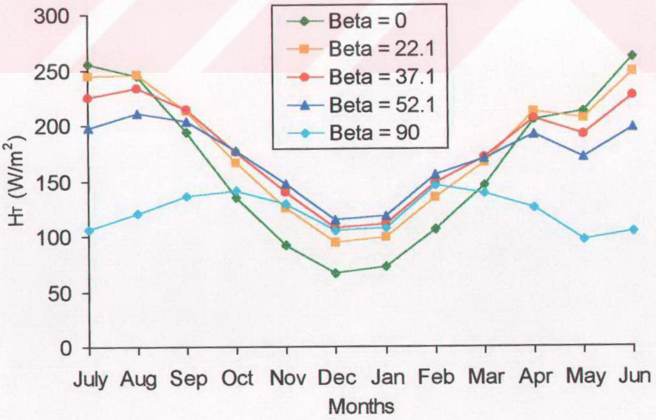


Figure 5.3. Variation of monthly average solar radiation on a collector surface with collector slope.

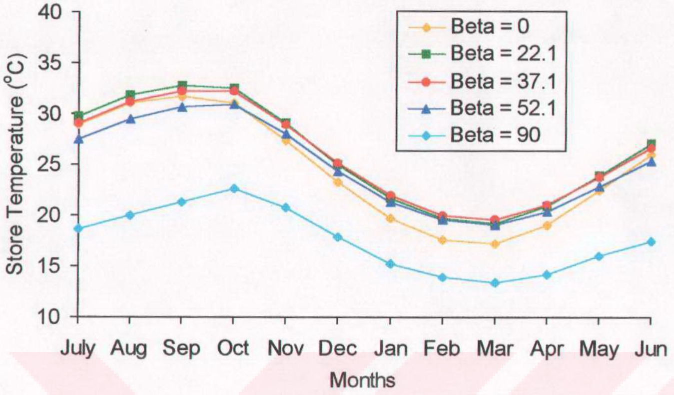


Figure 5.4. Effect of collector slope on the thermal storage temperature for one house (Gaziantep, one glass-cover black surface, A_c = 30 m^2, a = 5 m, Limestone)

Figure 5.5 shows the annually periodic variation of water temperature in the storage buried in four different earth types and for one house. The highest temperature occurs at the end of the summer season while the lowest temperature occurs at the end of the winter season. Energy charged to the storage by solar collectors throughout the summer season results in an increase in the temperature of the storage. Extraction of energy from the storage through the winter season results in a decrease in the storage temperature. It is seen also from the figure that highest temperatures occur in the storage surrounded with sand which has the lowest thermal conductivity while lowest temperatures occur in the storage surrounded with granite which has the highest thermal conductivity.

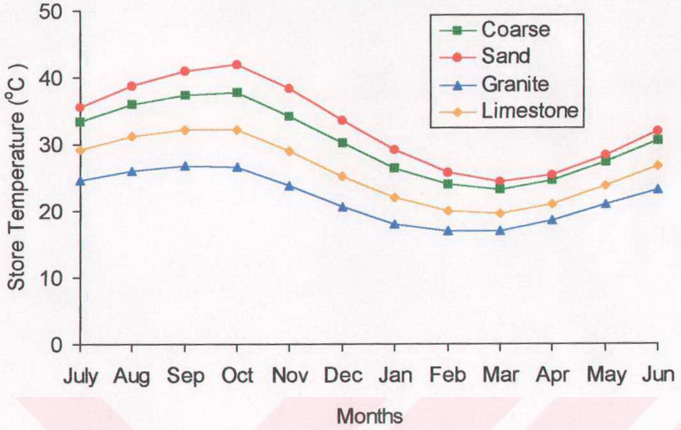


Figure 5.5. Annual variation of the water temperature in the storage for different geological structures (Gaziantep, one glass-cover black surface, $A_c = 30\ m^2$, a = 5 m, $\beta = 37.1°$)

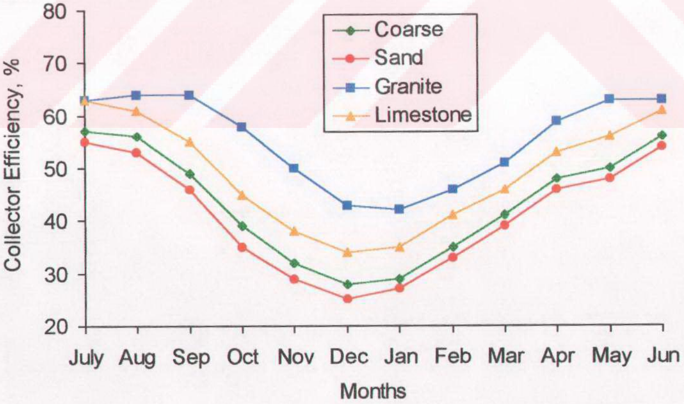


Figure 5.6. Annual variation of the collector efficiency for different geological structures (Gaziantep, one glass-cover black surface, $A_c = 30\ m^2$, a = 5 m, $\beta = 37.1°$)

Figure 5.6 shows the variation of the monthly average collector efficiency for four types of earth. The highest collector efficiency takes place for granite while the lowest efficiency is for sand. Figure 5.5 shows that the low temperatures of the storage buried in granite gives the highest collector efficiency, and the high temperatures in the storage surrounded with sand gives the lowest collector efficiency.

Annual periodic temperature of the storage embedded in limestone for several storage sizes considering one house is depicted in Figure 5.7. It is seen from this Figure that amplitude of the water temperature variation is higher for storages with smaller volume. Amplitude of the storage temperature decreases with increasing storage size. This figure also illustrates that small storage size gives a higher storage temperature at the end of the summer season while giving very low temperatures towards to the end of the winter months. The highest storage size gives the lowest annual average storage temperature while the lowest storage size give the highest.

Figure 5.8 shows the variation of the monthly average collector efficiency for several storage sizes. The annual mean collector efficiency was found to be largest for a storage radius of 7 m, and found to be smallest for a 3 m radius. Collector efficiency is directly proportional to the storage size. Because of increasing storage size, storage temperature decreases. Decreasing storage temperatures causes an increase in collector efficiency.

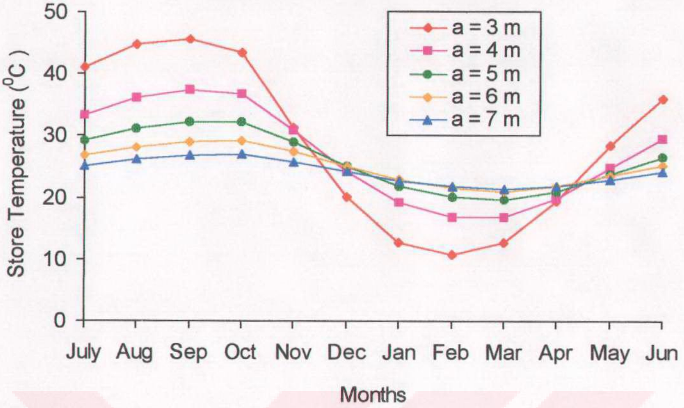


Figure 5.7. Annual variation of the water temperature in the storage buried in limestone for different storage sizes (Gaziantep, one glass-cover black surface, $A_c = 30$ m^2, $\beta = 37.1$ °)

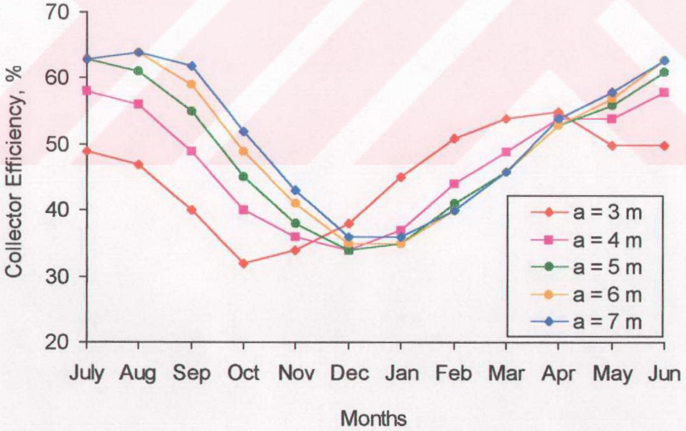


Figure 5.8. Annual variation of the collector efficiency for different storage sizes buried in limestone (Gaziantep, one glass-cover black surface, $A_c = 30$ m^2, $\beta = 37.1$ °)

Annual solar fraction for the storage with 5 m radius considering four different earth types is given in Figure 5.9 as a fraction of collector area. The figure demonstrates that the highest solar fraction occurs when the storage is surrounded with sand and minimum solar fraction is obtained with granite. It is seen from this figure that solar fraction increases with increasing collector area in a proportional manner as mentioned in İnallı et. al.[36]. Earths that have small thermal conductivity and thermal diffusivity give larger solar fraction.

Annual solar fraction versus collector area is given in Figure 5.10 to depict the effect of storage size. Small storage sizes give small annual solar fractions. This agrees with results from Lund and Östman[34] and İnallı et. al.[36]. Annual solar fraction increases with collector area. Storage size has only a small effect on the annual solar fraction. There are very small changes in the magnitude of the solar fraction for larger storage sizes, those having radius larger than 4 m.

Figures 5.11 and 5.12 illustrate variation of annual coefficient of performance of heat pump (COP) with collector surface area for different earth types, and storage sizes. It is seen from this Figure that annual COP of the heat pump is reasonably high. The COP increases with the collector area. It is observed that highest COP is obtained when the storage is buried in sand, lowest COP is obtained with granite. Figure 5.12 indicates that the smallest COP is obtained when the storage radius is 3 m. Storage size has small effect on the COP of the heat pump when radius of the storage is greater than 4 m.

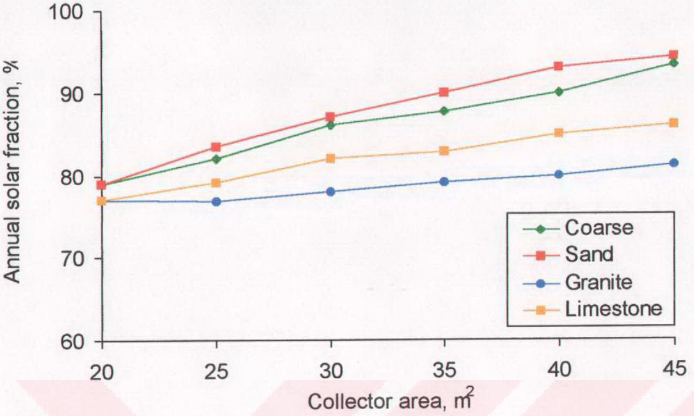


Figure 5.9. Variation of annual solar fraction for a storage with 5 m radius. (Gaziantep, one house, one glass-cover black surface, β = 37.1 °)

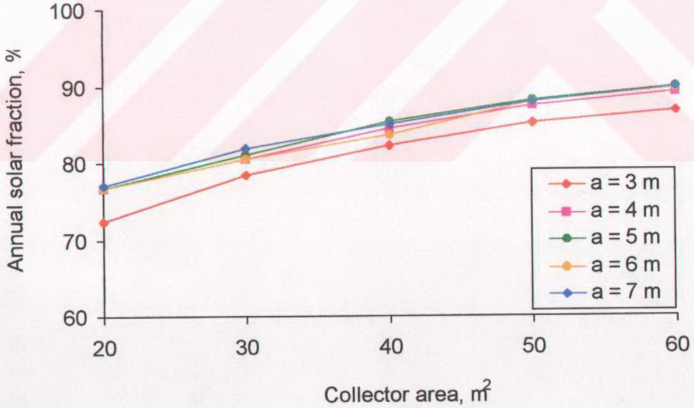


Figure 5.10. Annual solar fraction versus collector area for different storage sizes (Gaziantep, one house, one glass-cover black surface, β = 37.1 °, limestone)

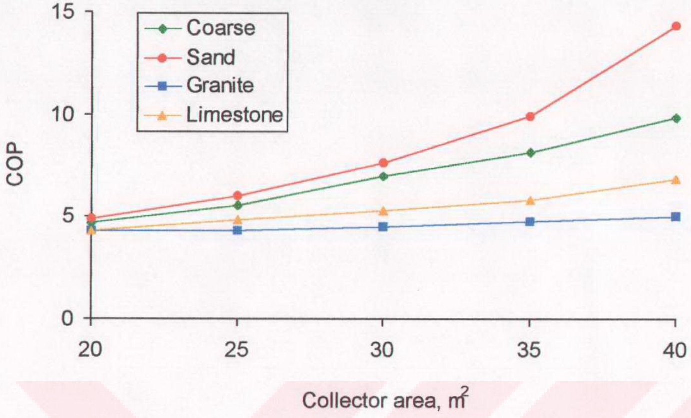


Figure 5.11. Annual coefficient of performance of the heat pump versus collector area for four different earth types (Gaziantep, one house, one glass-cover black surface, β = 37.1 °, a = 5 m)

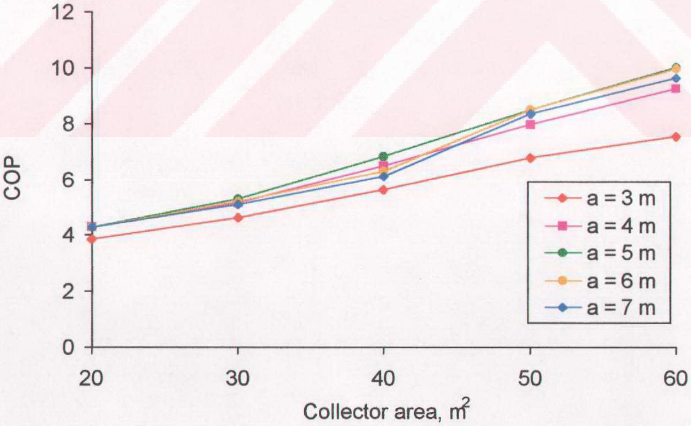


Figure 5.12. Annual coefficient of performance of the heat pump versus collector area for different storage sizes (Gaziantep, one house, one glass-cover black surface, β = 37.1 °, limestone)

The effect of collector slope on annual solar fraction and annual water temperature for storages surrounded with four different earth types are depicted in Figures 5.13 and 5.14 respectively. The highest solar fraction and water temperature takes place in the storages buried with sand for all collector slopes. Lowest values are obtained for granite. Maximum solar fraction occurs when the collector slope is equal to latitude angles for all earth types shown in Figure 5.13. Figure 5.14 illustrates that maximum solar fraction is obtained when the collector slope is equal to 30 degrees and this result is in agreement with that given in İnallı et. al.[36]

Figure 5.15 shows a variation of annually average storage water temperature with storage size. It is seen from the figure that the highest annual mean water temperature occurs in storage buried in sand while the lowest is for granite, and annual mean storage temperature decreases with increasing storage radius. The slope of the temperature change decreases with increasing storage radius.

Annual water temperature distribution in the storage embedded in limestone is given in Figure 5.16 using meteorological data for five cities located in Turkey. Lowest storage water temperatures occur with the meteorological data of İstanbul and the highest storage water temperatures occur for Gaziantep.

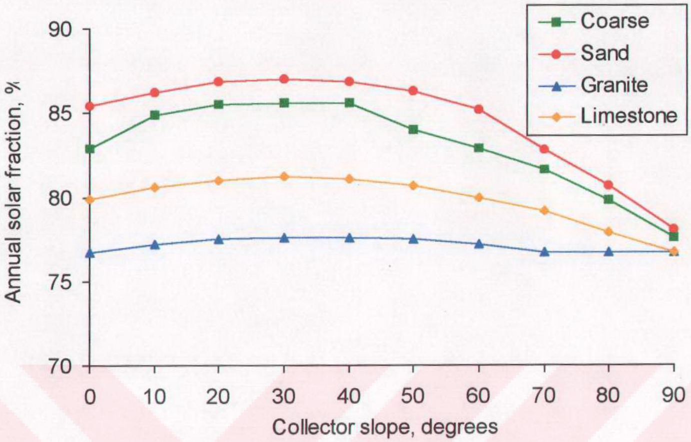


Figure 5.13. Annual solar fraction versus collector slope for a storage size with 5-m radius (Gaziantep, one house, one glass-cover black surface, A_c = 30 m^2)

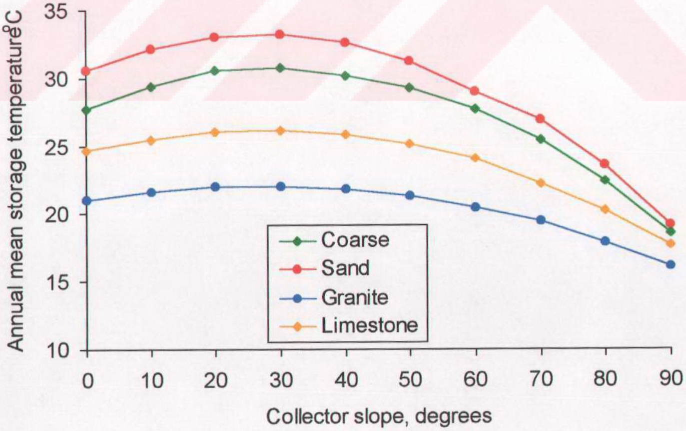


Figure 5.14. Annual solar fraction versus collector slope for a storage size with 5-m radius (Gaziantep, one house, one glass-cover black surface)

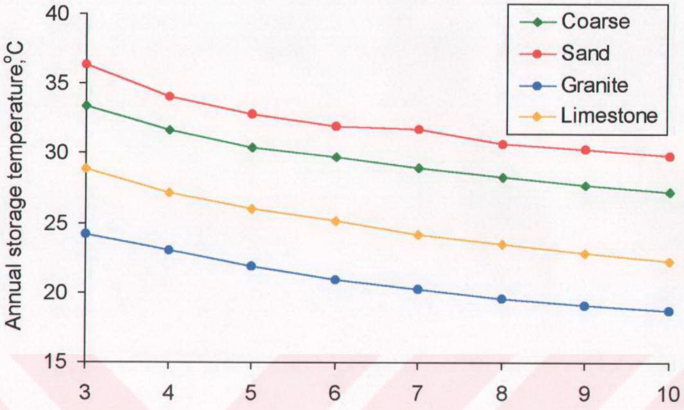


Figure 5.15. Annual water temperature in the storage versus storage radius for different earth types (Gaziantep, one house, one glass-cover black surface)

Total energy supplied to the heating system during the one year is solar energy and heat pump work. Solar energy supplied to the storage is partially lost into the surrounding earth and remaining part extracted for house heating. Annual energy balance of the thermal energy storage surrounded with limestone is presented in Figure 5.17 for several cities considering a system with one house. The lowest heat loss from the storage takes place from a storage in a system with İstanbul's meteorological data while the highest loss take place a storage in a system with Gaziantep's meteorological data. The highest solar energy fraction and the lowest heat pump work fraction occurs for İzmir while the lowest solar energy fraction and highest heat pump work fraction are for Elazığ. It is seen that the meteorological data has important effects on the annual energy fractions. Energy lost to the surrounding earth is approximately 15-25 percent, 75-85 percent being used for house heating for all cities considered.

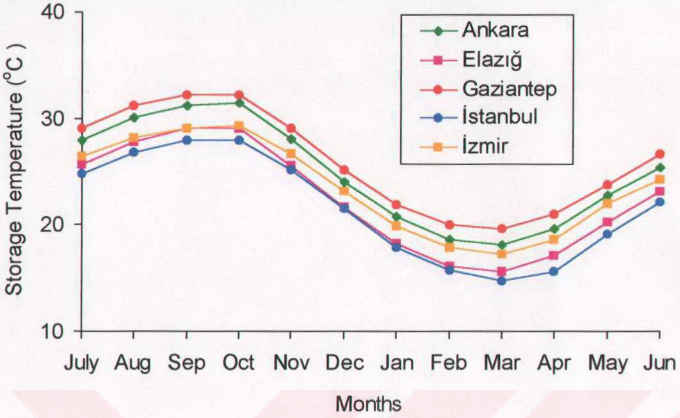


Figure 5.16. Annual variation of the water temperature in the storage embedded in Limestone (one house, one glass-cover black surface, $A_c = 30$ m^2, $\beta = \phi_L$, a = 5 m)

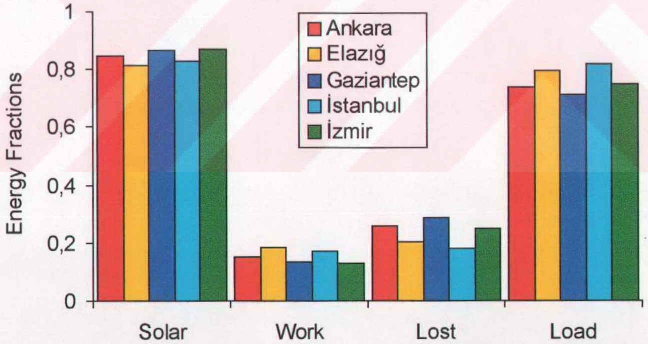


Figure 5.17. Annual energy fractions for different cities (one house, one glass-cover black surface, $A_c = 30$ m^2, $\beta = \phi_L$, a = 5 m, limestone)

Annual energy balances are given in Figure 5.18 for several storage sizes. It is observed from this Figure that the highest storage size gives the highest solar energy fraction and the lowest heat pump work fraction. Heat lost from the storage to surrounding earth increases and heat supplied to the house decreases with increasing storage size. It is seen that storage size has a small effect on the annual energy fractions.

Figure 5.19 shows the effect of earth type on the annual energy fractions for a thermal energy storage with a radius of 5 m. The highest solar energy fractions and house heat load fractions take place for a storage surrounded by sand. The lowest heat pump work fractions and heat loss fractions take place for a storage surrounded. for sand The lowest solar energy fractions and house heat load fractions take place for a storage surrounded by granite. The highest heat pump work fractions and heat loss fractions take place for granite. It is seen that type of earth has a small effect on the annual energy fractions.

Figures 5.20 through 5.25 are for a heating system with 100 houses. Figures 5.20 and 5.21 depict annual variation of water temperature and collector efficiency for a system with a storage located inside limestone and for different storage sizes considering a load consisting of 100 houses. It is observed that amplitude of the storage water temperature and the collector efficiency are larger for storages with smaller sizes. It is seen that the yearly average water temperature decreases and the yearly average collector efficiency increases when the system size is increased.

Monthly average value of the periodic storage temperature and collector efficiency are depicted in Figure 5.22 and 5.23 for 100 houses and for four different types of earth. It is observed from these figures that the amplitudes of the storage temperature and the collector efficiency increase when the system size increases. Yearly averages of these values are larger than those for small systems consisting of a single house. Effects of earth types on the storage temperatures and collector efficiency decreases when the system size is increased.

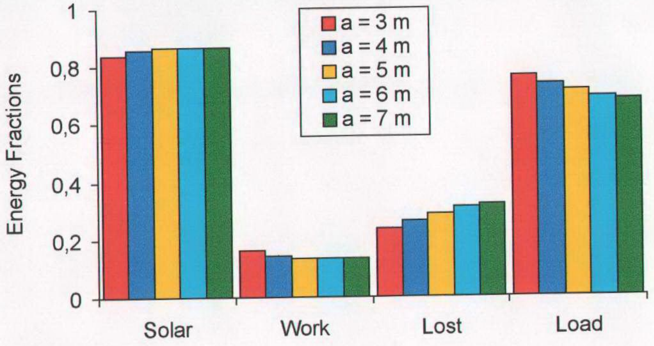


Figure 5.18. Effect of storage size on annual energy fractions (Gaziantep, one house, one glass-cover black surface, $A_c = 30$ m^2, $\beta = 37.1$ °, limestone)

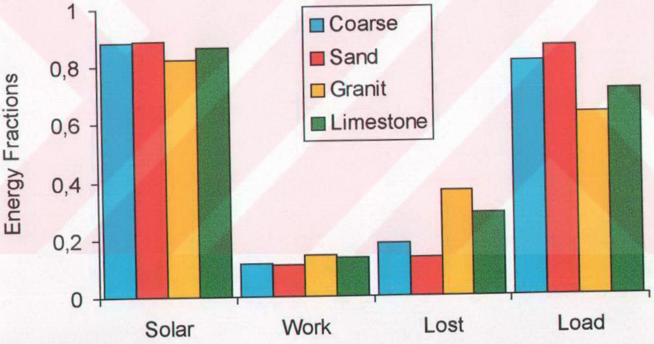


Figure 5.19. Effect of type of earth on annual energy fractions (Gaziantep, one house, one glass-cover black surface, $A_c = 30$ m^2, $\beta = 37.1$ °, a = 5 m, limestone)

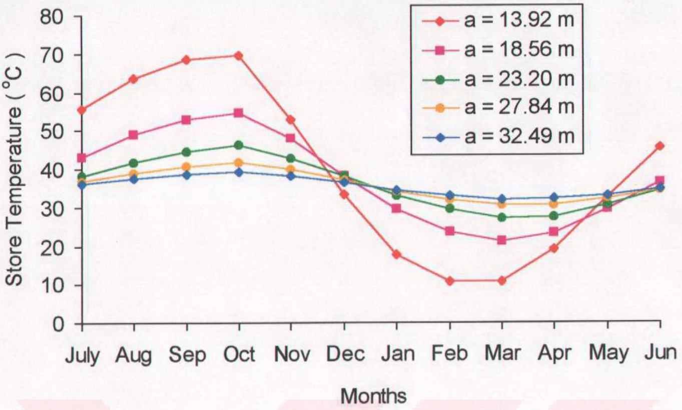


Figure 5.20. Annual variation of the water temperature in the storage vessel for different storage radii buried in limestone (Gaziantep, 100 houses, one glass-cover black surface, A_c = 30 m^2, β = 37.1 °)

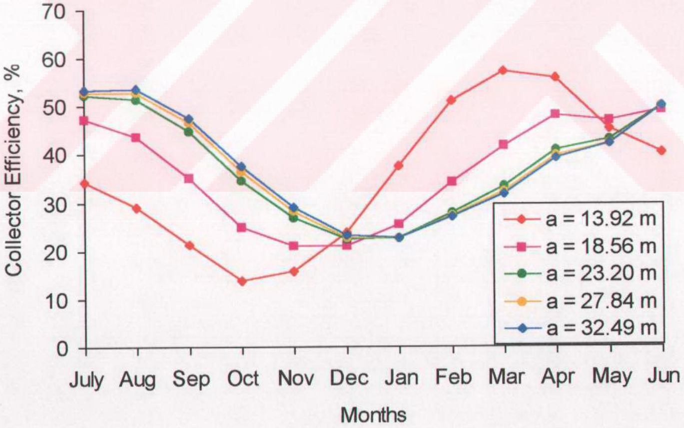


Figure 5.21. Annual variation of the collector efficiency for different storage size buried in limestone (Gaziantep, 100 houses, one glass-cover black surface, A_c = 30 m^2/house, β = 37.1 °)

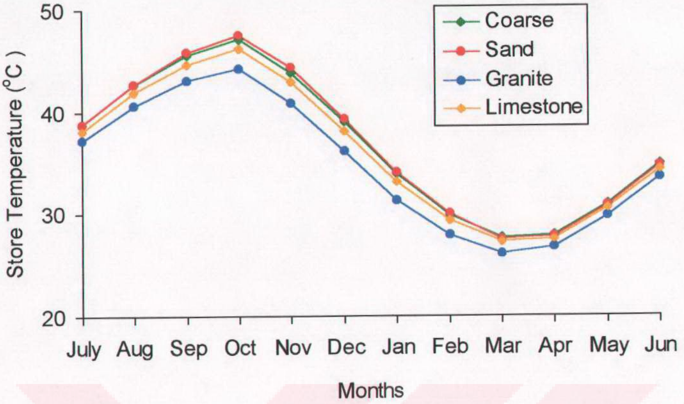


Figure 5.22. Annual variation of the storage water for different types of earth (Gaziantep, 100 houses, one glass-cover black surface, A_c = 30 m^2/house, β = 37.1 °, a = 23.20 m)

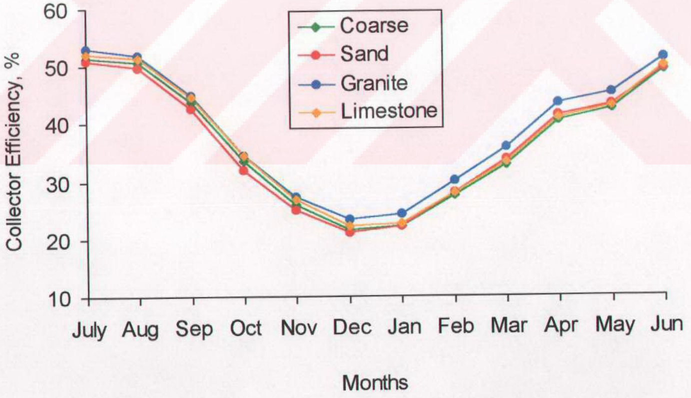


Figure 5.23. Annual variation of the collector efficiency for four different types of earth (Gaziantep, 100 houses, one glass-cover black surface, A_c = 30 m^2/house, β = 37.1 °, a = 23.2 m)

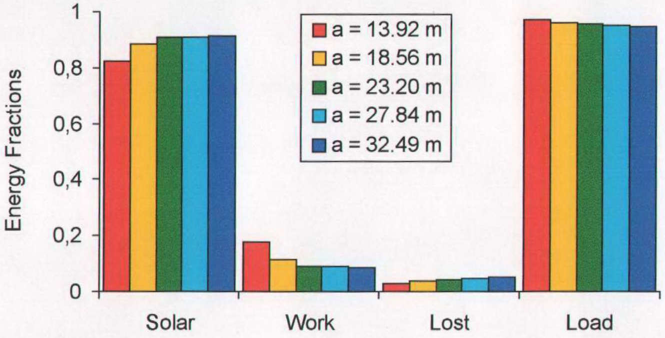


Figure 5.24. Effect of storage size on annual energy fractions for 100 houses (Gaziantep, one glass-cover black surface A_c = 30 m^2/house, β = 37.1 °, limestone)

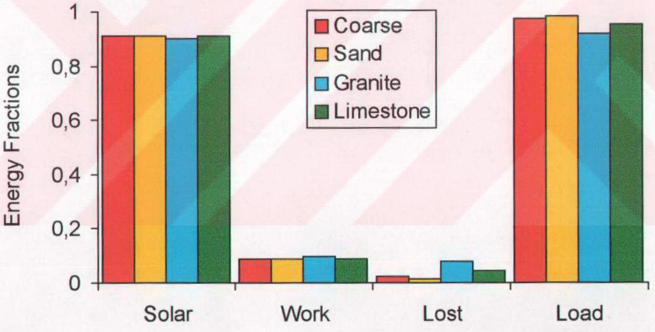


Figure 5.25. Effect of earth types on annual energy fractions for 100 houses(Gaziantep, one glass-cover black surface, A_c = 30 m^2/house, β = 37.1 °)

Fractions of the energy quantities are compared with Figures 5.24 and 5.25 for different storage volumes and types of earth for a system with a load consisting of 100 houses. It is observed that system performance increases with system size, that is, a high system size gives lower heat loss fractions and a higher fractions for the house heat supply.

5.4 RESULTS FOR HEATING SYSTEMS WITH HEMISPHERICAL STORAGES

The simulation program SOLARSTORE was executed to obtain performance of a heating system with seasonal underground hemispherical energy storage. The results obtained from the execution of the SOLARSTORE for one and 100 houses are given in graphical forms and discussed in this section.

The results are given for two cases; one for biot number is equal to zero and another for biot number larger than zero. Figures 5.26-58 are given for the case of zero Biot number and for an angle of θ_0=85 degrees. Figures 5.26 and 5.27 show the effect of earth types on the water temperature in the storage and collector efficiency for a storage tank which has a volume of 420 m^3 and radius of 6 m. It is seen from the figure that the water temperature in the storage is quite high. The highest amplitude of the annual water temperature is obtained when the storage is embedded in sand, while the lowest amplitude is obtained while the storage is buried in granite. The highest water temperature and the lowest collector efficiency are obtained when the storage is buried in sand. The lowest storage temperature and highest collector efficiency occur when it is buried in granite.

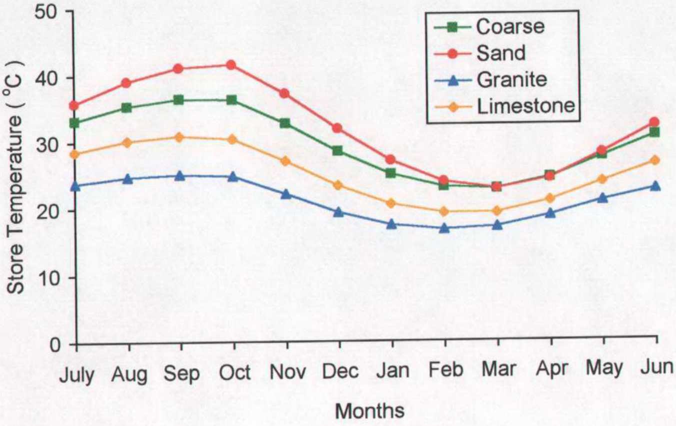


Figure 5. 26. Effect of earth types on the water temperature in the storage for different geological structures (Gaziantep, one glass-cover black surface, $A_c = 30$ m^2, R = 6 m, $\beta = 37.1$ °,Biot=0.0)

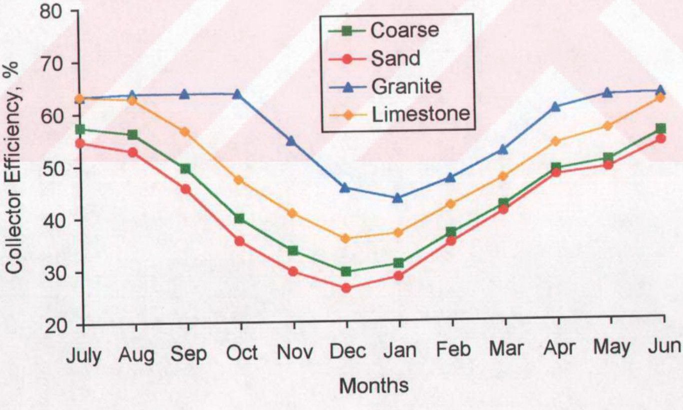


Figure 5.27. Effect of earth types on the collector efficiency for different geological structures (Gaziantep, one house, one glass-cover black surface, $A_c = 30$ m^2, R = 6 m, $\beta = 37.1$ °, , Biot=0.0)

Effects of storage sizes on the water temperature and collector efficiency for a storage buried in limestone are depicted in figure 5.28 and 5.29. The Figure 5.28 illustrates that the smallest storage gives the highest variations in storage temperature while the highest storage gives lowest variation in a season. There is a small change in annual water temperature in the storage when the storage size has a radius of 8 m. It is observed from the figures that the small storage size gives higher storage temperature at summer months and lower temperature at winter months. On the contrary, the small storage gives lower collector efficiency at summer season and higher efficiency at winter season. A phase difference between maximum storage temperature and maximum efficiency exists for all the storage sizes.

Figures 5.30-5.32 show temperature distribution in the surrounding earth. Annual temperature distribution in earth for different distances from the storage buried in limestone is given in figure 5.30. It is seen from the figure that the amplitude of annual ground temperature decreases when the distance increases, and the ground temperature remains constant if the distance from the storage is increased beyond a certain distance. Figure 5.31 indicates temperature distribution versus distances from the storage in ground for four months. It is observed the highest temperature occurs at around the storage occurs in summer season but the lowest ground temperature occurs in winter season. At the same time, ground temperature remains constant when the distance increases. This condition illustrates that the storage looses energy through summer season and receives from the ground in winter season. Figure 5.32 demonstrates ground temperature distribution versus distance from the storage buried in limestone for different storage sizes in October. It is observed that temperature variation in earth decreases when the storage size and distance from the storage increases.

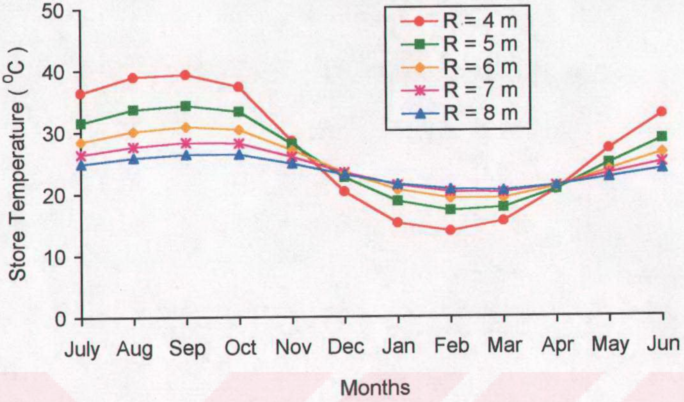


Figure 5.28. Effect of earth types on the water temperature in the storage buried in limestone (Gaziantep, one house, one glass-cover black surface, $A_c = 30\ m^2$, $\beta = 37.1\ °$,Biot=0.0)

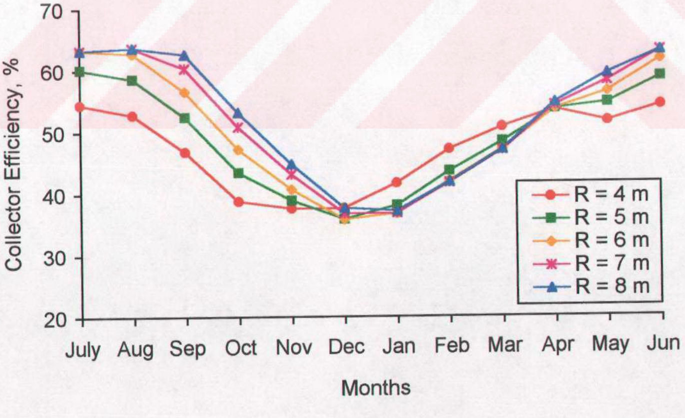


Figure 5.29. Effect of earth types on the collector efficiency for storage buried in limestone (Gaziantep, one glass-cover black surface, $A_c = 30\ m^2$, $\beta = 37.1\ °$, Biot=0.0)

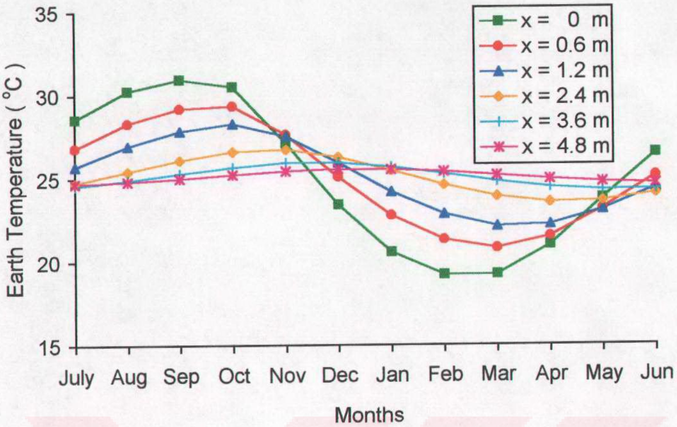


Figure 5. 30. Annual temperature distribution in earth for different distances (Gaziantep, one house, one glass-cover black surface, $A_c = 30\ m^2$, R = 6 m, $\beta = 37.1$ °, Limestone, Biot=0.0)

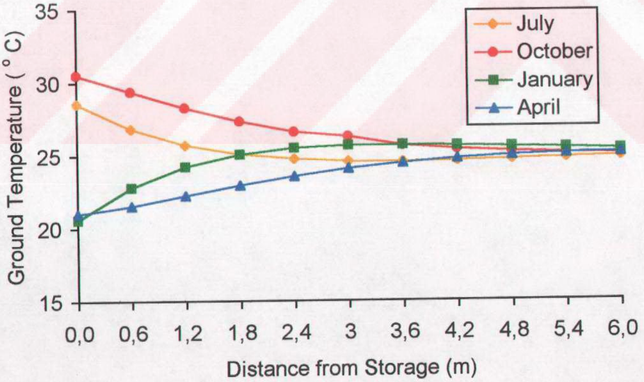


Figure 5.31. Temperature distribution versus distances from the storage in earth for four months (Gaziantep, one glass-cover black surface, $A_c = 30\ m^2$, R = 6 m, $\beta = 37.1$ °, Limestone, Biot=0.0)

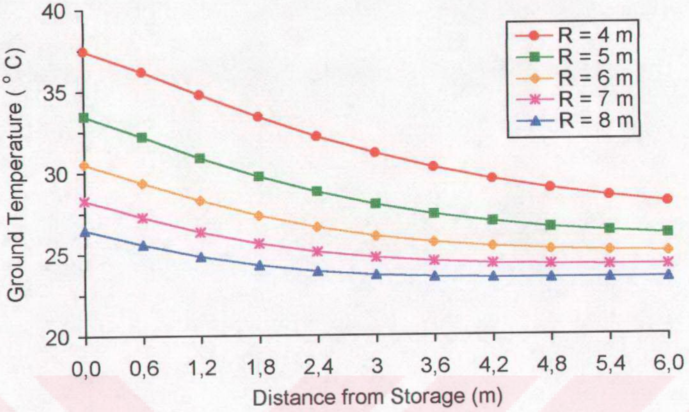


Figure 5. 32. Ground temperature distribution versus distances from the storage for different storage sizes (Gaziantep, one glass-cover black surface, A_c = 30 m^2, β = 37.1 °, Limestone, Biot=0.0)

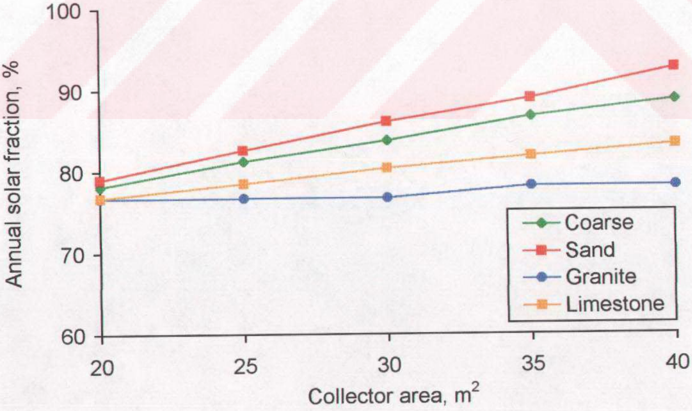


Figure 5.33. Variation of annual solar fraction for a storage with 6 m radius. (Gaziantep, one house, one glass-cover black surface, β = 37.1 °, Biot=0.0)

Annual variation of solar fraction versus collector surface area for a storage with 6 m radius is shown in figure 5.33. It is seen from the figure that solar fraction is quite high. The figure shows that the lowest solar fraction is obtained when the storage is buried in granite while the highest is obtained for sand. Increasing collector surface area increases solar fraction and makes the effect of earth type much more clearer.

Figures 5.34 and 5.35 show the effect of collector slope on the annual storage temperature and solar fraction for a storage size of 6 m radius. The lowest annual water temperature in the storage and solar fraction occurs for granite while the highest temperature and solar fraction occurs for sand. Maximum storage temperature and solar fraction takes place for a collector slope between 30 and 40 °C.

Annual water temperature in the storage versus storage radius for different earth types is given in figure 5.36. The Figure illustrates that annual mean water temperature in the storage decreases with increasing storage size. It is clear that high storage size requires large storage surface area which increases heat losses from storage to the surrounding ground thus increasing heat losses and decreasing the water temperature in the storage.

Effect of collector surface area on annual coefficient of performance of heat pump versus collector area for different earth types and storage sizes are depicted in Figures 5.37 and 5.38 respectively. The Figures show that the coefficient of the performance of heat pump is quite high. The effect of collector area on COP of the heat pump decreases when the collector area decreases. COP is approximately 4.3 with a collector area of 20 m^2. Figure 5.38 illustrates that the effect of storage size on the annual COP of heat pump when collector area is small.

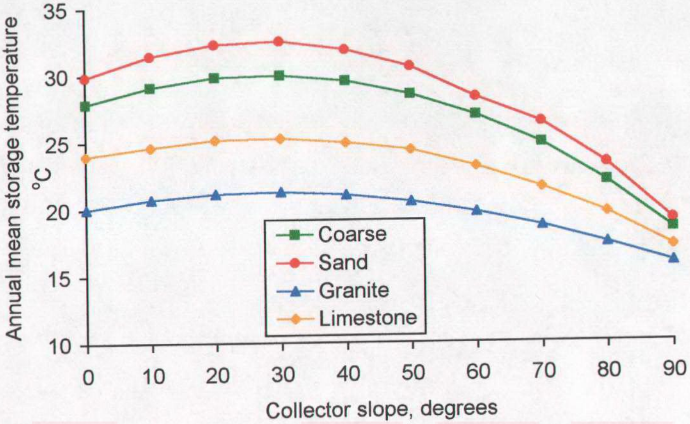


Figure 5.34. Annual solar fraction versus collector slope for a storage size with 6-m radius (Gaziantep, one house, one glass-cover black surface, A_c = 30 m^2, Biot=0.0)

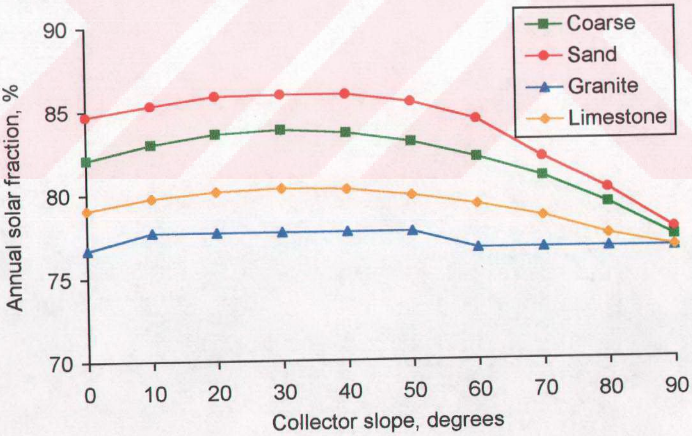


Figure 5.35. Annual solar fraction versus collector slope for a storage size with 6-m radius (Gaziantep, one house, one glass-cover black surface, A_c = 30 m^2 , Biot=0.0)

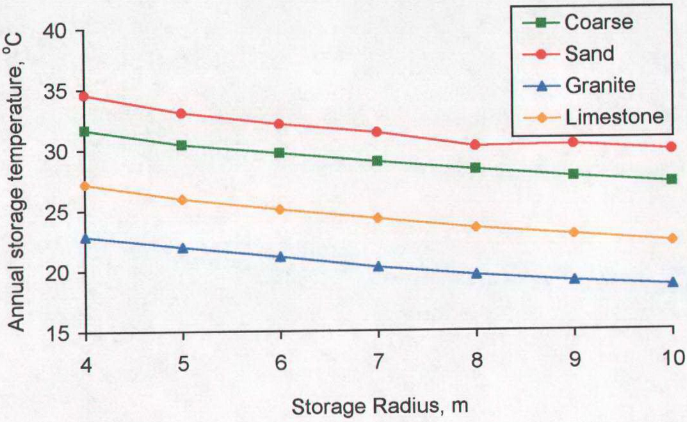


Figure 5.36. Annual water temperature in the storage versus storage radius for four different earth types (Gaziantep, one glass-cover black surface, A_c = 30 m^2, β = 37.1 °, Biot=0.0)

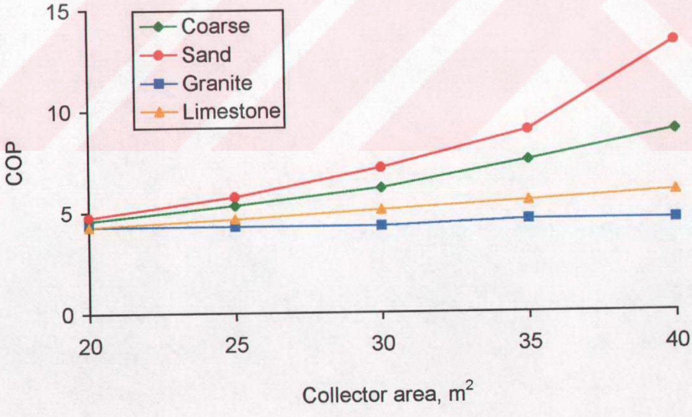


Figure 5.37. Annual coefficient of performance of the heat pump versus collector area for four different earth types (Gaziantep, one house, one glass-cover black surface, β = 37.1 °, R = 6 m, Biot=0.0)

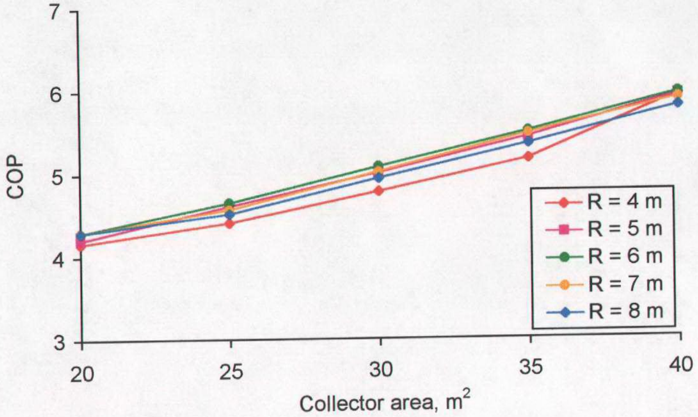


Figure 5.38. Annual coefficient of performance of the heat pump versus collector area for different storage sizes (Gaziantep, one house, one glass-cover black surface, β = 37.1 °, limestone, Biot=0.0)

Figure 5.39 and 5.40 summarize the effect of earth type and storage size on annual energy fractions for a heating system with one house. It is seen from the figures that solar energy fractions are quite high while heat pump work fractions are quite low

Figures 5.41-5.58 are depicted for the heating system with 100 houses. System performance parameters obtained from execution of the simulation program SOLARSTORE are discussed for large system size. Figures 5.41 and 5.42 show annual variations of the water temperature in the storage and collector efficiency for four different earth types. These figures illustrate that the annual amplitude of the storage temperature and collector efficiency increases when the system size increases. When the annual water temperature in the storage for a system of 100 houses is compared with a system of one house, it is seen from the Figure 5.41 that range of the amplitude of water temperature in the storage is about 20 °C and 25 °C and is about 10 °C and 20 °C of water temperature variation given in Figure 5.26. Also the Figures 5.41 and 5.42 indicate that the effects of earth types on the storage temperature and collector efficiency decreasing when the system size is increased from one house to 100 houses. For large system, the storage temperature approaches each other for different earth types.

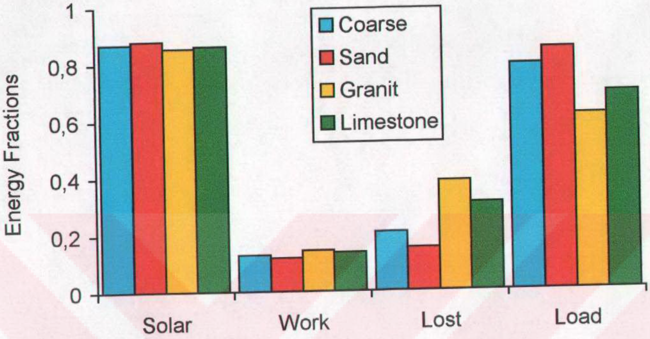


Figure 5.39. Effect of type of earth on annual energy fractions (Gaziantep, one house, one glass-cover black surface, A_c = 30 m^2, β = 37.1 °, R = 6 m, Biot=0.0)

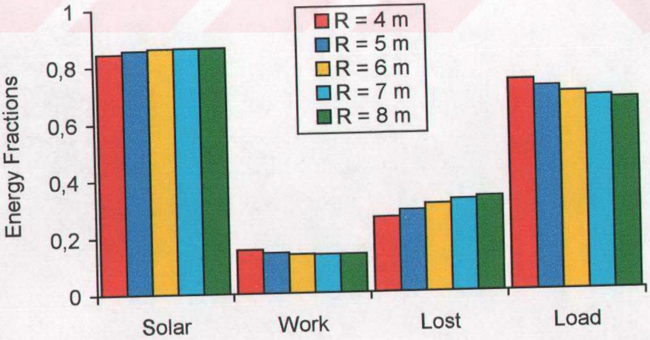


Figure 5.40. Effect of storage size on annual energy fractions (Gaziantep, one house, one glass-cover black surface, A_c = 30 m^2, β = 37.1 °, limestone, Biot=0.0)

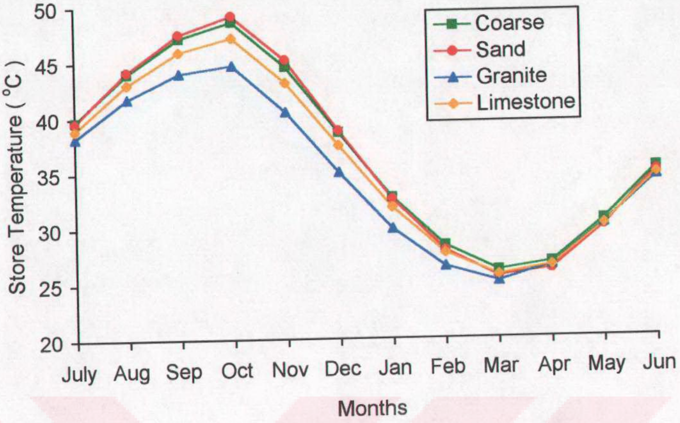


Figure 5. 41. Annual variation of the water temperature in the storage for different geological structures for 100 houses (Gaziantep, one glass-cover black surface, $A_c = 30\ m^2$, R = 27.84 m, $\beta = 37.1\ °$, Biot=0.0)

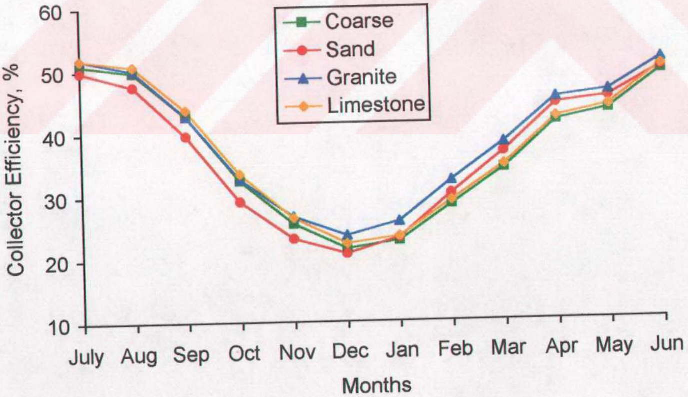


Figure 5.42. Annual variation of the collector efficiency for different geological structures for 100 houses (Gaziantep, one glass-cover black surface, $A_c = 30\ m^2$, R = 27.84 m, $\beta = 37.1\ °$, Biot=0.0)

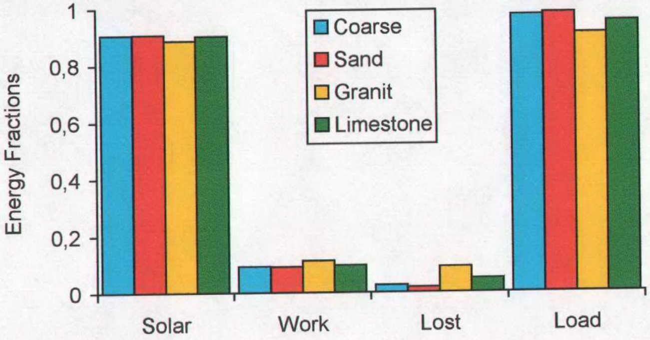


Figure 5.43. Effect of type of earth on annual energy fractions (Gaziantep, 100 houses, one glass-cover black surface, A_c = 30 m^2, β = 37.1 °, R = 27.84 m, Biot=0.0)

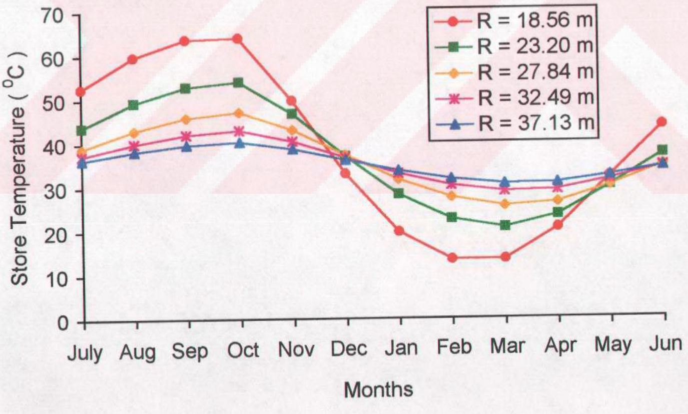


Figure 5.44. Annual variation of the water temperature in the storage buried in limestone for different storage sizes (Gaziantep, one glass-cover black surface, A_c = 30 m^2, β = 37.1 °, 100 houses, Biot=0.0)

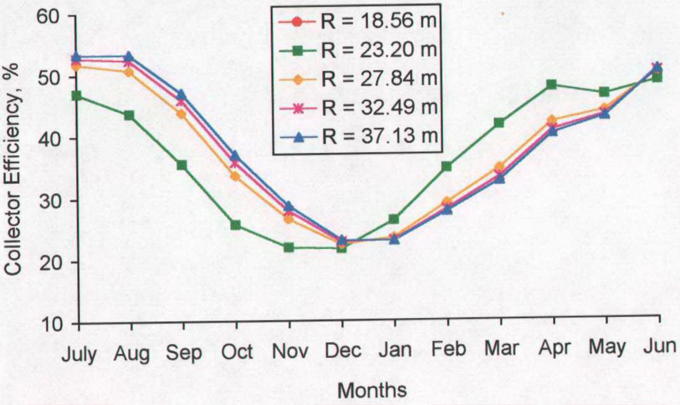


Figure 5.45. Annual variation of the collector efficiency in the storage buried in limestone for different storage sizes (Gaziantep, one glass-cover black surface, A_c = 30 m^2, β = 37.1 °, 100 houses, Biot=0.0)

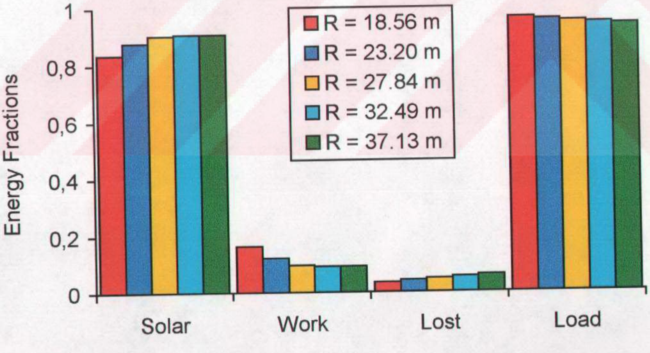


Figure 5.46. Effect of storage size on annual energy fractions (Gaziantep, one house, one glass-cover black surface, A_c = 30 m^2, β = 37.1 °, limestone, Biot=0.0)

Effect of earth type on annual energy fraction is given in figure 5.43 for large system with 100 houses and storage volume of 413 m^3/house. The figure points out that solar energy and heat load fraction increases while heat pump work and heat lost fractions decreases when the system size is increased.

Figure 5.44 and 5.45 demonstrate annual variation of water temperature in the storage and collector efficiency for heating system with 100 houses. It is seen from the figures that the amplitude of the water temperature in the storage increases with the system size. The highest storage temperature and low collector efficiency are obtained at summer season. The lowest storage temperature obtained at winter season.

Effects of collector area and storage size on the energy fractions for a large system size are depicted in Figures 5.49 and 5.50. It is seen from the Figure 5.50 that solar energy and heat pump work fractions are approximately 80 and 20 percent for a system with a collector area of 20 m^2 and storage radius 20 m. For the system low heat lost and high heat load fractions are obtained.

Effects of storage size on annual energy fractions for heating system with 100 houses is depicted in Figure 5.46. When the Figure 5.40 for small heating system is compared with the Figure 5.46 for large heating system, It is seen from these figures that solar and load fractions increase, heat pump work and lost fractions decrease with increasing system size.

Figures 5.47 and 5.48 show annual variation of water temperature in the storage and collector efficiency for four different store sizes. It is seen from the Figure 5.47 that the storage size with a radius of 20 m for large system with 100 houses is suitable when the storage is exactly isolated. But energy loss occurs from top of the storage. So, larger store size is necessary for the heating system.

Figure 5.51 illustrates annual variation of the water temperature in the storage buried in limestone for different collector area. Higher the collector area gives higher water temperature.

Figures 5.52 and 5.53 indicate annual variation of water temperaure in the storage with a radius of 20 m and collector efficiency for different geological structures for 100 houses. It is observed that effect of earth types decreases when the system size increases.

Figures 5.54 and 5.55 show annual solar fraction and coefficient of performance of the heat pump versus collector area for four different earth types for a heating system with a storage radius of 20 m. Annual solar fraction and COP are quite high. Solar fraction and COP are about 78 percent and 4.6 respectively. It is seen from the figures that the effect of collector area on the solar fraction and COP is very small when the system size is increased.

Figure 5.56 denotes the effects of earth types on annual energy fractions for a heating system with a storage radius of 20 m and a collector area of 20 m^2. It is observed that solar energy and heat pump with fractions are about 80 and 20 percent respectively, and also heat lost and load fractions approach to zero and 100 percent.

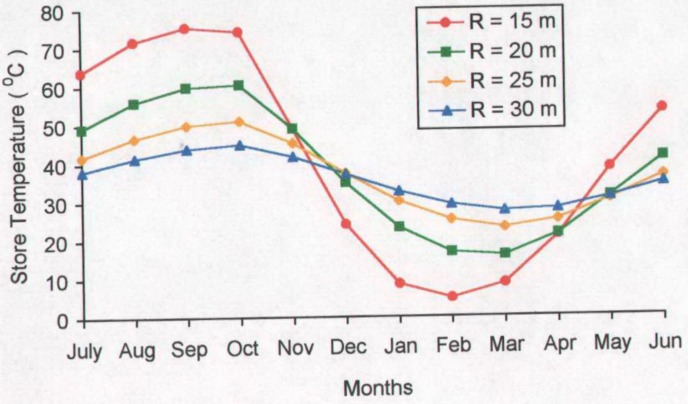


Figure 5.47. Annual variation of the water temperature in the storage buried in limestone for different storage sizes (Gaziantep, one glass-cover black surface, $A_c = 30$ m^2, $\beta = 37.1$ °, 100 houses, Biot=0.0)

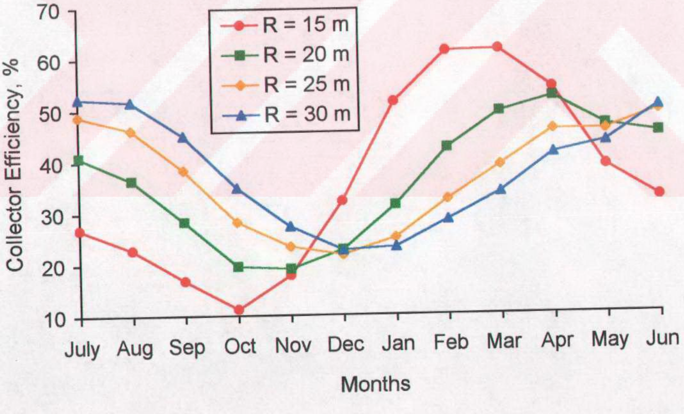


Figure 5.48. Annual variation of the collector efficiency for different storage sizes (Gaziantep, one glass-cover black surface, $A_c = 30$ m^2, $\beta = 37.1$ °, 100 houses, limestone, Biot=0.0)

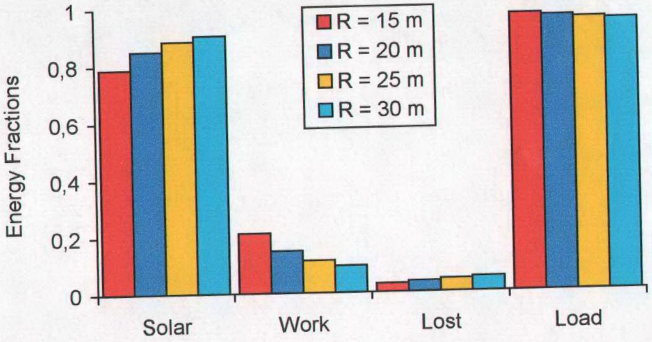


Figure 5.49. Effect of storage size on annual energy fractions (Gaziantep, 100 houses, one glass-cover black surface, A_c = 30 m^2, β = 37.1 °, limestone, Biot=0.0)

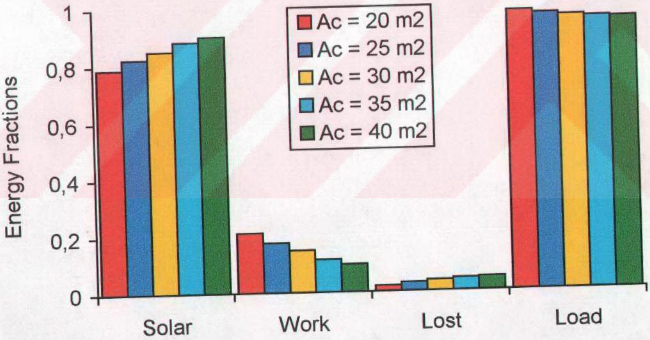


Figure 5.50. Effect of storage size on annual energy fractions (Gaziantep, 100 houses, one glass-cover black surface, R = 20 m, β = 37.1 °, limestone, Biot=0.0)

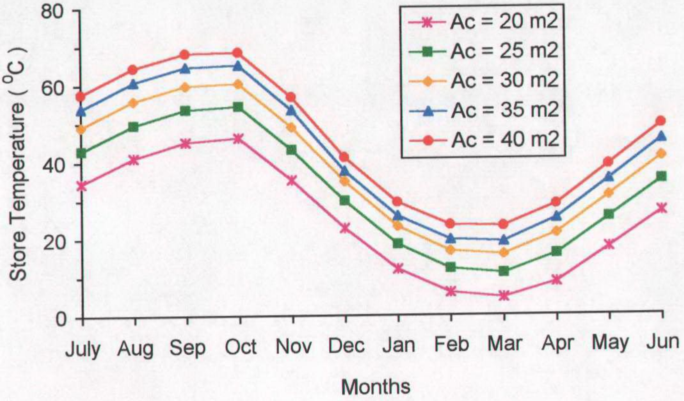


Figure 5.51. Annual variation of the water temperature in the storage buried in limestone for different collector area (Gaziantep, one glass-cover black surface, $\beta = 37.1$ °, R = 20 m, 100 houses, Biot=0.0)

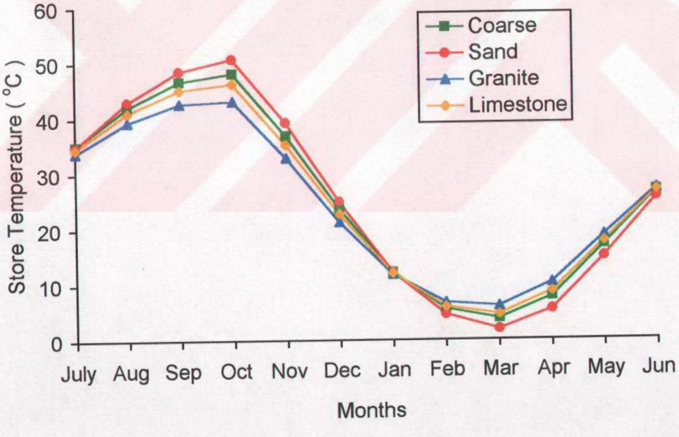


Figure 5. 52. Annual variation of the water temperature in the storage for different geological structures for 100 houses (Gaziantep, one glass-cover black surface, $A_c = 20\ m^2$, R = 20 m, $\beta = 37.1$ °, Biot=0.0)

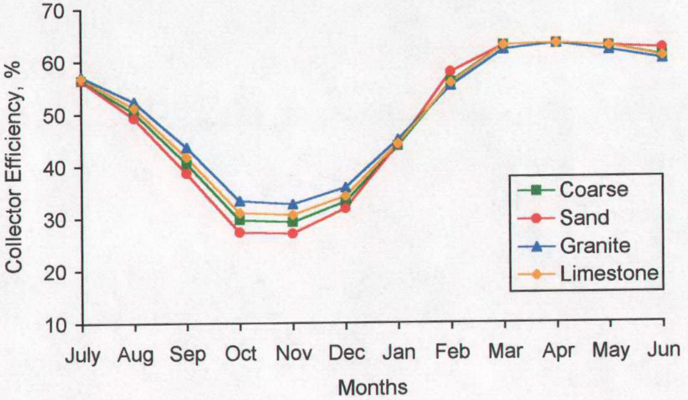


Figure 5.53. Annual variation of the collector efficiency for different geological structures for 100 houses (Gaziantep, one glass-cover black surface, $A_c = 20\ m^2$, R = 20 m, β = 37.1 °, Biot=0.0)

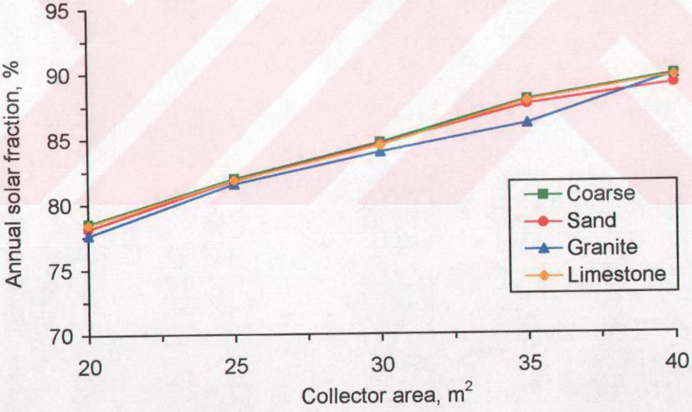


Figure 5.54. Variation of annual solar fraction for a storage with 20 m radius. (Gaziantep, one glass-cover black surface, β = 37.1 °, 100 houses, Biot=0.0)

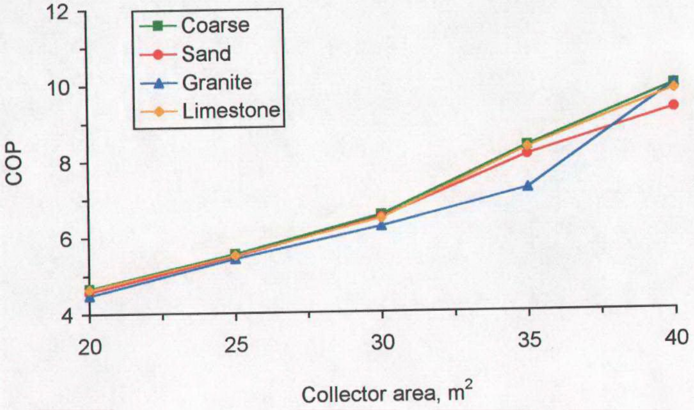


Figure 5.55. Annual coefficient of performance of the heat pump versus collector area for four different earth types (Gaziantep, 100 house, one glass-cover black surface, $\beta = 37.1$ °, R = 20 m, , Biot=0.0)

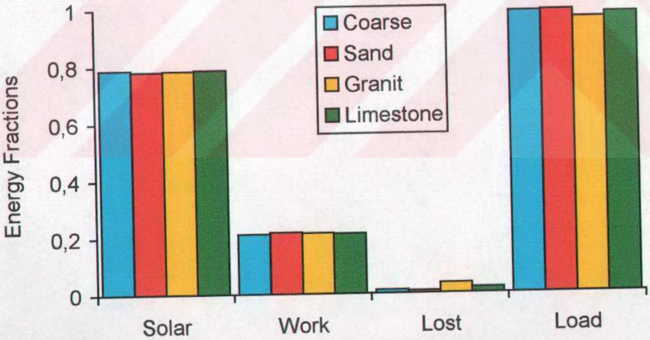


Figure 5.56. Effect of type of earth on annual energy fractions (Gaziantep, 100 houses, one glass-cover black surface, $A_c = 20$ m², $\beta = 37.1$ °, R = 20 m, Biot=0.0)

Annual variation of ground temperature and annual ground temperature distributions for varying distances from the storage for four months are given in Figures 5.57 and 5.58. It is seen from the Figure 5.57 that the amplitude of annual earth temperature is the highest near the storage. The amplitude decreases while the distance from the storage increases. Figure 5.58 shows that the highest ground temperature occurs at the end of summer season and the lowest temperature takes place at the end of winter season. The ground temperature stays constant for heating system with a storage of radius of 20 m and 100 houses after a distance of 4 meters.

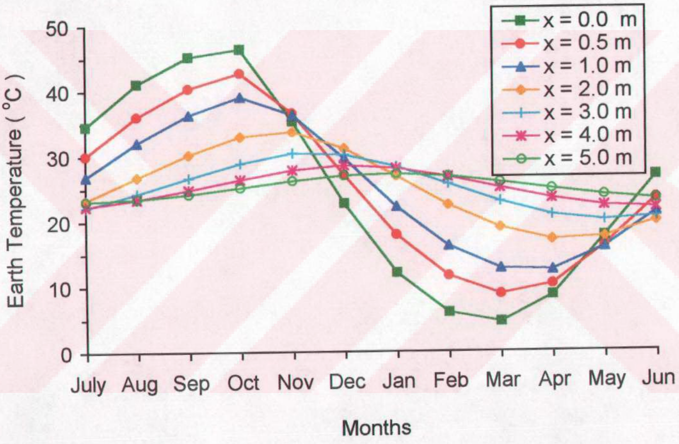


Figure 5. 57. Annual variation of the ground temperature for different distance from storage (Gaziantep, one glass-cover black surface, A_c = 20 m^2, R = 20 m, β = 37.1 °, Limestone, 100 houses, Biot=0.0)

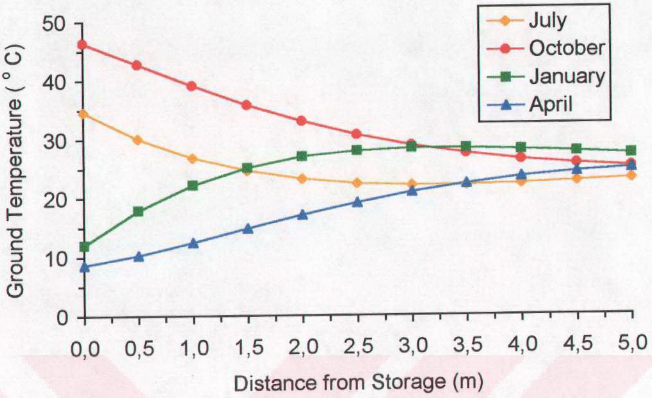


Figure 5. 58. Annual variation of the ground temperature in the storage for different months (Gaziantep, one glass-cover black surface, $A_c = 20$ m^2, R = 20 m, $\beta = 37.1$ °, Limestone, 100 houses, Biot=0.0)

Figure 5.59-5.85 are depicted to study effect of Biot number on storage and surrounding ground temperature, solar fraction, coefficient of performance of heat pump and energy fractions.

Figure 5.59 shows annual variations of water temperature in the storage embedded in limestone for different Biot number and for 100 houses. It is observed that higher the Biot number gives higher storage temperature. A small difference in storage temperature is observed when the Biot number is changed from 8.31 to 12.5. This Figure illustrates that the storage size with a radius of 25 m is enough for a system size of 100 houses.

The effect of Biot number on annual water temperature in the storage for four different earth types is depicted in Figures 5.60-5.62. It is seen from the Figure 5.61 that the effect of earth types on storage temperature is small when the Biot number is taken as zero. There is a high temperature difference in summer while a small difference in winter months. Figures 5.61 and 5.62 show storage temperature for Biot number of 8.31 and 25 respectively. The effect of earth types on storage temperature increases when Biot number is increased. This corresponds to a decrease in isolation thickness at top of the energy storage. In this case, these three figures show that the water temperature in the storage decrease when the Biot number is increased, while annual amplitude of the storage temperature increases with decreasing Biot number. The storage temperature in winter months decreases faster than the summer months.

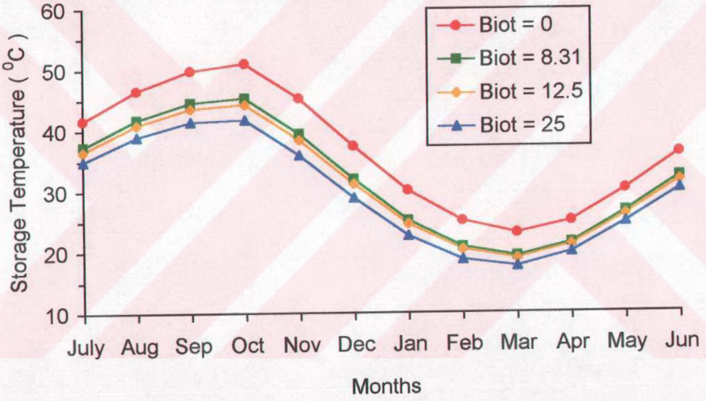


Figure 5.59. Effect of Biot numbers on annual water temperature in the storage buried in limestone for 100 houses(Gaziantep, one glass-cover black surface, $\beta = 37.1$ °, R = 25 m)

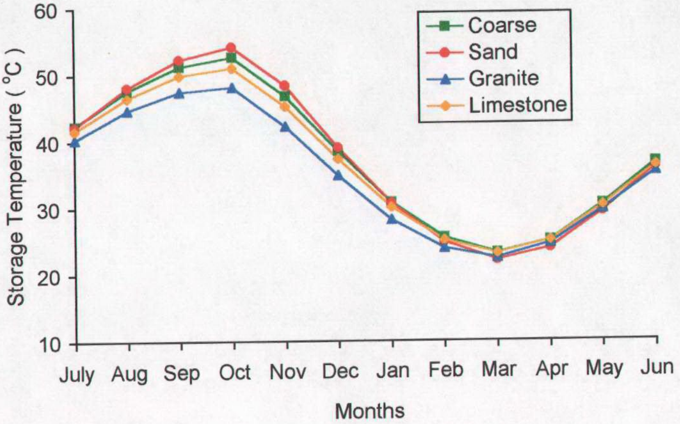


Figure 5. 60. Annual variation of the water temperature in the storage for different geological structures for zero Biot number (Gaziantep, 100 houses, one glass-cover black surface, A_c = 30 m^2, R = 25 m, β = 37.1°, Biot=0)

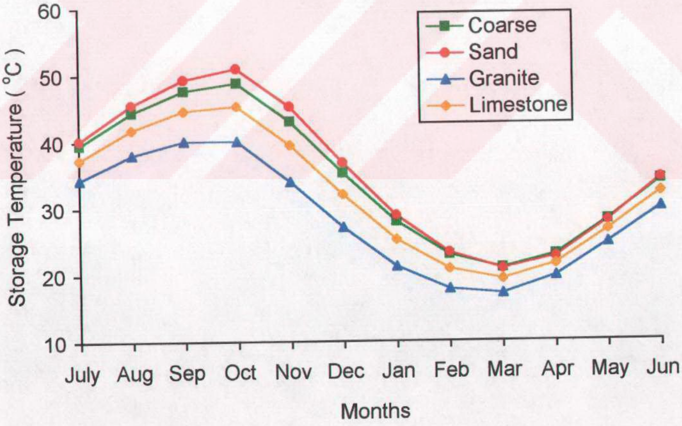


Figure 5. 61. Annual variation of the water temperature in the storage for different geological structures for Biot number equal to 8.31 (Gaziantep, 100 houses one glass-cover black surface, A_c = 30 m^2, R = 25 m, β = 37.1 °, z = 3 m)

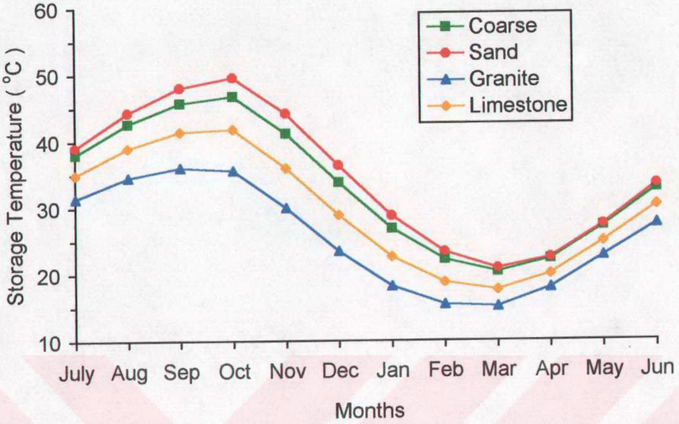


Figure 5. 62. Annual variation of the water temperature in the storage for different geological structures for Biot number equal to 25 (Gaziantep, 100 houses one glass-cover black surface, $A_c = 30\ m^2$, $R = 25$ m, $\beta = 37.1$ °, $z = 1$ m, Biot=25.0)

Figure 5.63, and 5.64 indicate annual ground temperature distribution versus distance from storage surface for October January and four different earth types at zero degree polar angle. While the largest temperature distribution in ground takes place when the storage is surrounded with sand, the smallest temperature distribution occurs with granite. It is seen from the Figure 5.63 that ground temperature decreases from the storage surface to 2 meters away. After 2 meters, temperature change in the ground decreases. But it is seen from the figure 5.64 that earth temperature increases rapidly up to a distance of 2 meters for a winter month of January.

Figures 5.65 and 5.66 illustrate annual variation of ground temperature at 1 and 5 meters for different polar angles. The highest annual earth temperature occurs when the polar angle is taken as 83.1°. The smallest annual temperature distribution in limestone occurs when the polar angle is taken 0°. Annual amplitude of earth temperature for a distance of 1 meter away from the storage is larger than the distance of 5 meters. This shows that amplitude of earth temperature decreases with the distance away.

Ground temperature distribution versus polar angle for different distances from the storage for September is depicted in Figure 5.67. Earth temperature decreases with the distances away from the storage. It is also seen from the figure that earth temperature at different locations increases when the polar angle is increased from the 0 to 83.1° in September.

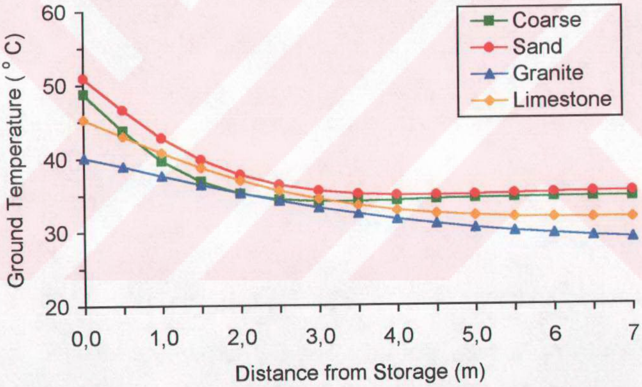


Figure 5. 63. Variation of the ground temperature versus distances from the storage for October (Gaziantep, 100 houses, one glass-cover black surface, $A_c = 30\ m^2$, $R = 25$ m, $\beta = 37.1$ °, Limestone, $z = 3$ m, Biot=8.31, $\theta = 0°$)

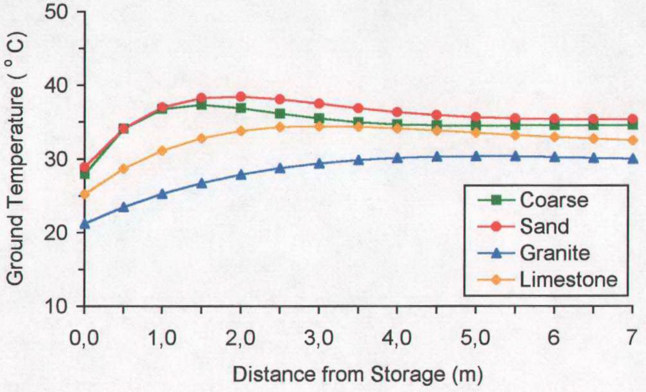


Figure 5. 64. Variation of the ground temperature versus distance from the storage for January (Gaziantep, 100 houses, one glass-cover black surface, $A_c = 30\ m^2$, R = 25 m, $\beta = 37.1$ °, Limestone, Biot=8.31, $\theta = 0°$)

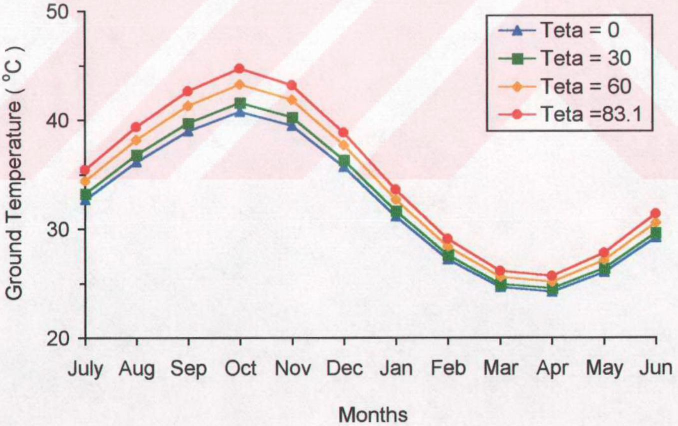


Figure 5. 65. Annual variation of ground temperature at 1 m away from storage for four different angles (Gaziantep, 100 houses, one glass-cover black surface, $A_c = 30\ m^2$, R = 25 m, $\beta = 37.1$ °, Biot = 8.31, Limestone)

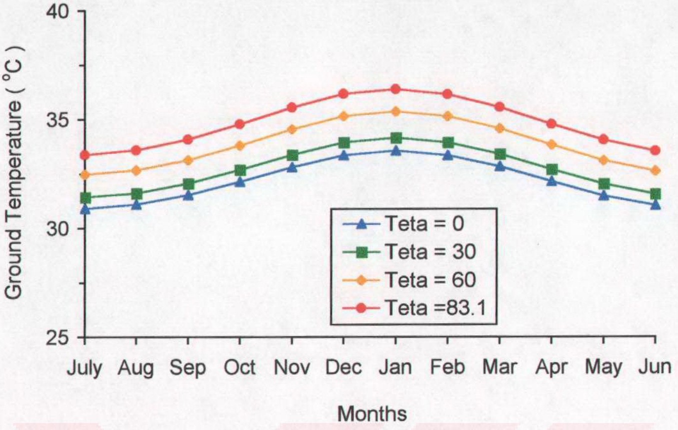


Figure 5. 66. Annual variation of ground temperature at 5 m away from storage for four different angles (Gaziantep, 100 houses, one glass-cover black surface, A_c = 30 m^2, R = 25 m, β = 37.1 °, Biot = 8.31, Limestone)

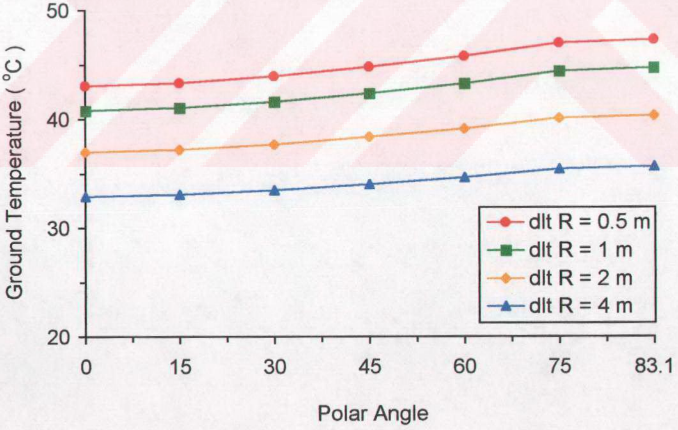


Figure 5. 67. Variation of ground temperature versus polar angle for September and different distances from storage (Gaziantep, 100 houses, one glass-cover black surface, A_c = 30 m^2, R = 25 m, β = 37.1°, Biot = 8.31, Limestone)

Figure 5.68,-5.70 show temperature distribution versus distance from the storage for four different months, and a polar angle of zero, 30, and 60 degrees respectively. The highest temperature in earth near the storage is obtained in October. Earth temperature decreases in summer season when distance from the storage increases. But, earth temperature increases with increasing the distance from the storage in winter months. In this case, increase in the distance from the storage results in a small change in earth temperature, i.e. earth temperature approaches to a asymptotic value at 30 °C. Maximum temperature variations takes place near the storage. These three figures shows that there is a small change in temperature distributions in earth with polar angle.

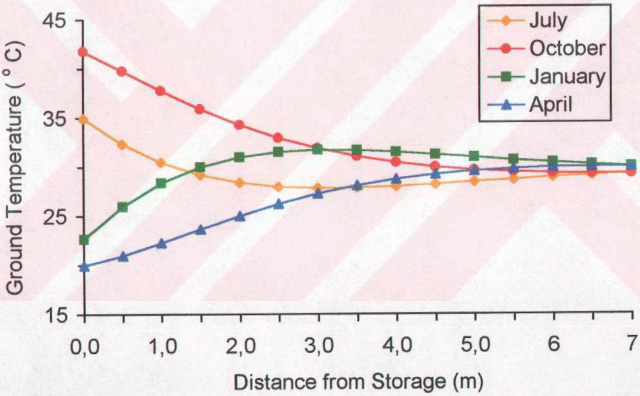


Figure 5. 68. Variation of ground temperature versus distance from the storage for different months and a polar angle of 0°(Gaziantep, 100 houses, one glass-cover black surface, A_c = 30 m^2, R = 25 m, β = 37.1°, Limestone, Biot=25)

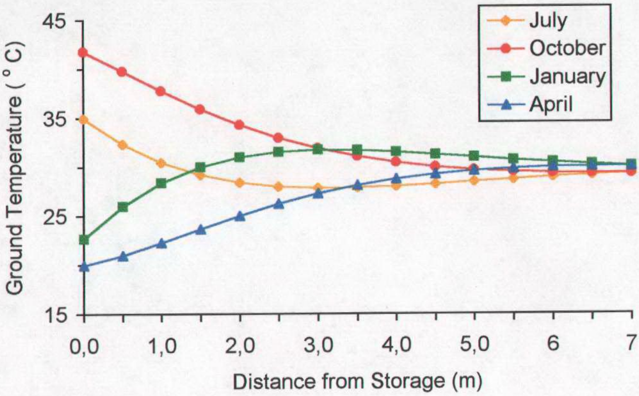


Figure 5. 69. Annual variation of the ground temperature versus distance from the storage for different months and a polar angle of 30° (Gaziantep, 100 houses, one glass-cover black surface, A_c = 30 m^2, R = 25 m, β = 37.1°, Limestone, Biot=25)

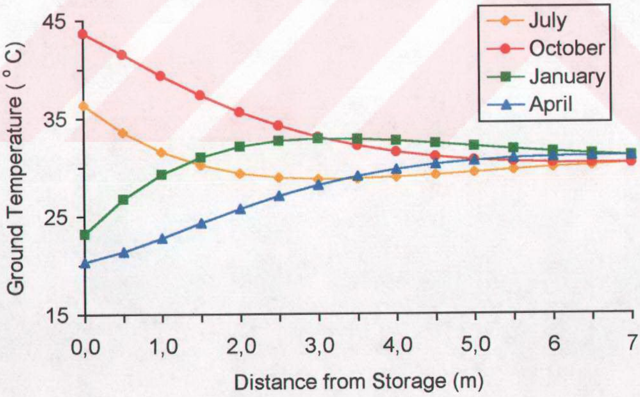


Figure 5. 70. Variation of ground temperature versus distance from the storage for different months and a polar angle of 60°(Gaziantep, 100 houses, one glass-cover black surface, A_c = 30 m^2, R = 25 m, β = 37.1 °, Limestone, Biot=25)

Figures 5.71 and 5.72 show temperature distribution in earth for October and January. It is observed that the highest ground temperature is obtained when Biot number is taken as zero while the lowest temperature is obtained for Biot number of 25. High Biot number gives low ground temperature and low Biot number gives high ground temperature. Earth temperature around the storage changes rapidly. It is seen from Figure 5.71 that ground temperature is high in summer season because solar energy is charged to the storage. The storage and surrounding ground exchanges the charged energy. It is also seen from Figure 5.72 that ground temperature near the storage is low in January. The storage temperature is low in January because of extraction of heat by heat pump. Biot number has higher effect on the storage and ground temperature.

Variation of ground temperature for October and January versus distance from the storage for four different polar angles is given in Figures 5.73 and 5.74. There is a high temperature near the ground surface where polar angle is equal to 87.7 degrees. Earth temperature changes rapidly up to 2-3 meters from the energy storage. After the distances of 2-3 meters, temperature approaches to a value of 28 degree centigrade. There is a small change in ground temperature with respect to the polar angles.

The effects of Biot number on annual ground temperature for different distance from the storage is depicted in Figures 5.75-5.77. Annual amplitude of the ground temperature decreases when the distance is increased away from the storage. While the highest and lowest temperature occurs in October and March, respectively at the storage surface. They shift to after several months, when the distance is away from the storage. These three figures illustrate that the highest annual storage and ground temperature is obtained when the Biot number is taken as zero, and the lowest annual storage and ground temperature distribution take place when Biot number is taken as 25. It is observed that annual storage and ground temperature decreases with increasing Biot number.

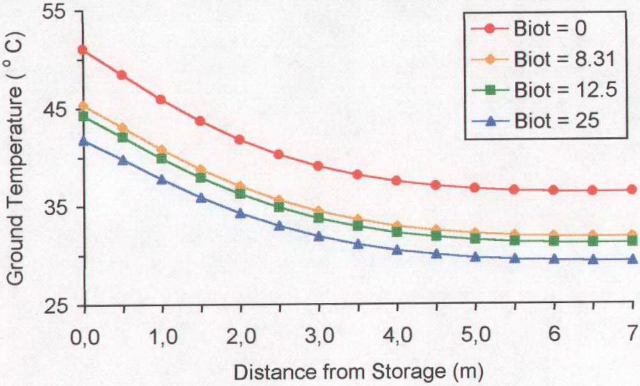


Figure 5. 71.Ground temperature distribution for October versus distance from the storage for four different Biot numbers (Gaziantep, 100 houses, one glass-cover black surface, A_c = 30 m^2, R = 25 m, β = 37.1 °, θ = 0°, limestone)

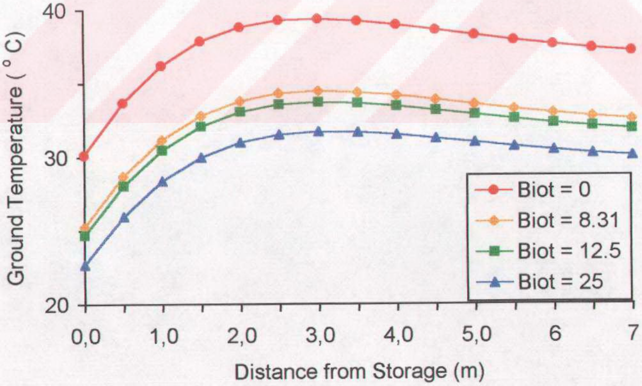


Figure 5. 72.Variation of ground temperature for January versus distance from the storage for four different Biot numbers (Gaziantep, 100 houses, one glass-cover black surface, A_c = 30 m^2, R = 25 m, β = 37.1°, θ =0°, limestone)

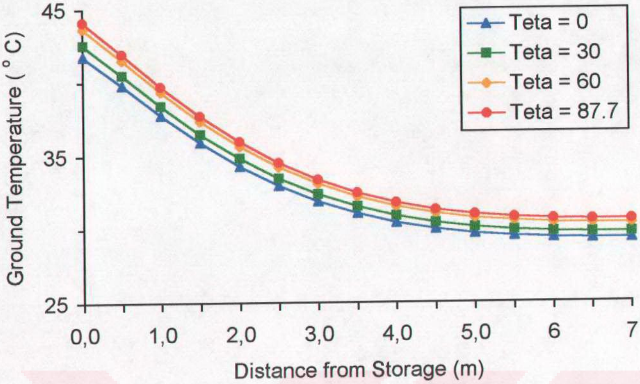


Figure 5. 73. Variation of ground temperature for October versus distance from the storage for different polar angles (Gaziantep, 100 houses, one glass-cover black surface, $A_c = 30\ m^2$, $R = 25$ m, $\beta = 37.1°$, limestone, Biot = 25)

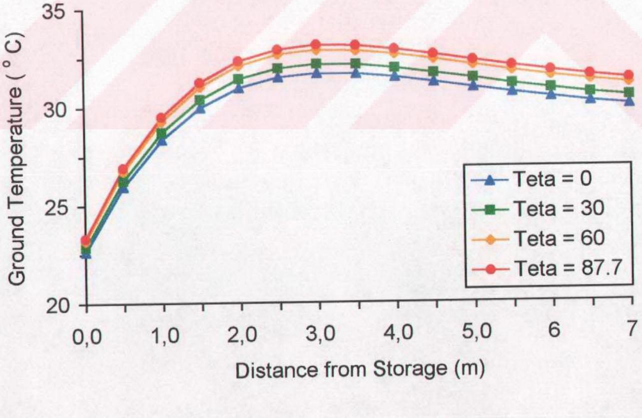


Figure 5. 74. Variation of ground temperature for January versus distance from the storages for different polar angles for January (Gaziantep, 100 houses, one glass-cover black surface, $A_c = 30\ m^2$, $R = 25$ m, $\beta = 37.1°$, limestone, Biot = 25)

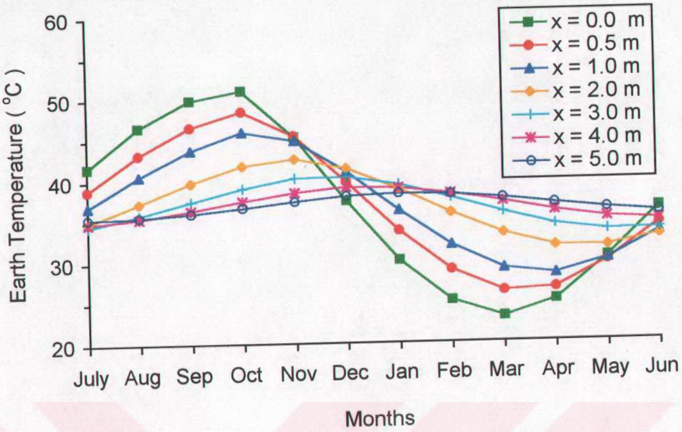


Figure 5. 75. Effect of Biot number on annual ground temperature for different distances from storage (Gaziantep, 100 houses, one glass-cover black surface, A_c = 30 m^2, R = 25 m, β = 37.1 °, Limestone, Biot=0, θ = 0)

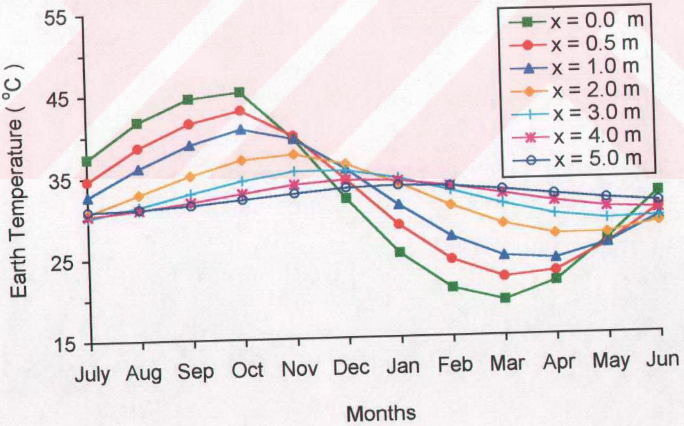


Figure 5. 76. Effect of Biot number on annual ground temperature for different distance from storage (Gaziantep, 100 houses, one glass-cover black surface, A_c = 30 m^2, R = 25 m, β = 37.1 °, Limestone, θ = 0, Biot = 8.31)

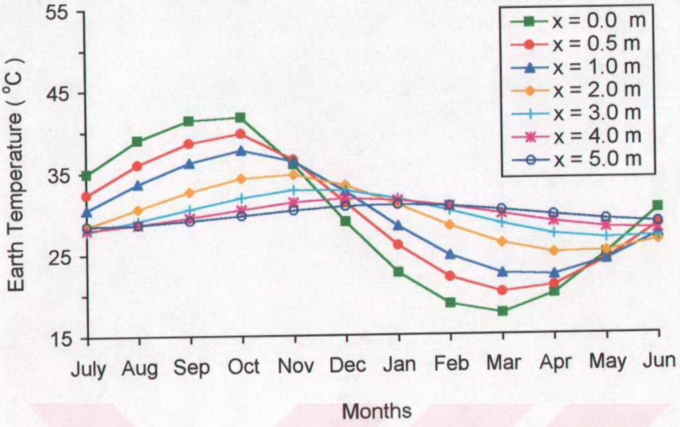


Figure 5. 77. Effect of Biot number on annual ground temperature for different distance from storage (Gaziantep, 100 houses, one glass-cover black surface, $A_c = 30\ m^2$, R = 25 m, $\beta = 37.1$ °, Limestone, $\theta = 0$, Biot = 25)

Annual solar fraction and annual coefficient of performance of heat pump for the Biot number of 25 considering four different earth types as given in Figures 5.78 and 5.79. Maximum annual solar fraction and heat pump COP occur when the storage is buried in sand and minimum in granite.

Effect of Biot number on the solar fraction and heat pump COP are depicted in Figures 5.80 and 5.81. It is seen from the figures that the lowest and highest solar fraction and heat pump COP are obtained when the Biot number is taken as 25 and zero respectively. There is a small difference between the Biot number values of 8.31 and 12.5. Effect of Biot number is high on the solar fraction but solar fraction values are high for the heating system at these conditions.

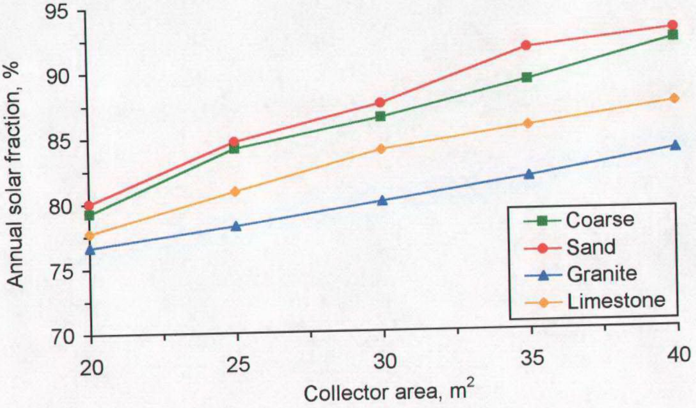


Figure 5.78. Variation of annual solar fraction for a storage of 25 m radius. (Gaziantep, 100 houses, one glass-cover black surface, β = 37.1 °, Biot=25)

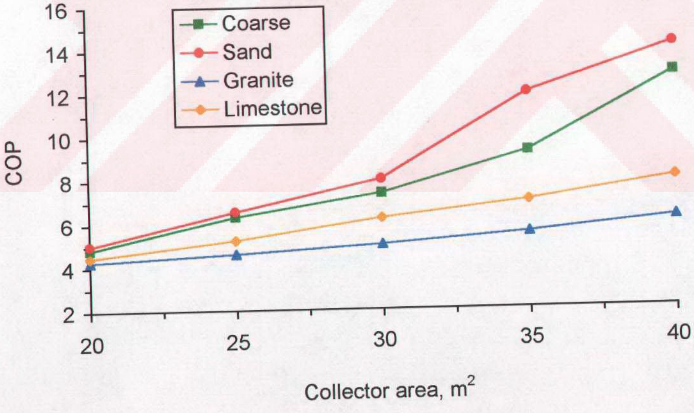


Figure 5.79. Annual coefficient of performance of the heat pump versus collector area for four different earth types (Gaziantep, 100 houses, one glass-cover black surface, β = 37.1 °, R = 25 m, , Biot=25)

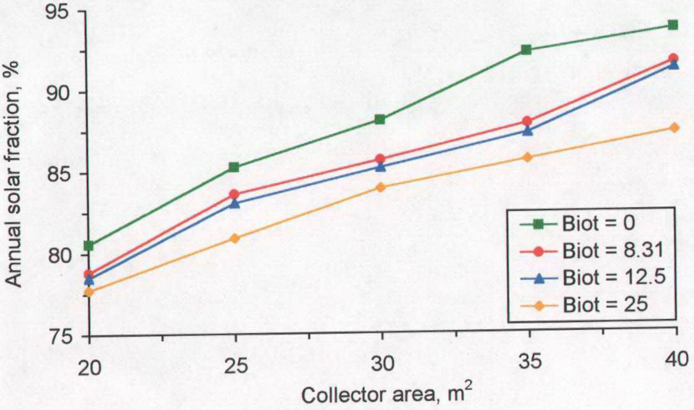


Figure 5.80. Variation of annual solar fraction for a storage of 25 m radius. (Gaziantep, 100 houses, one glass-cover black surface, β=37.1, Limestone)

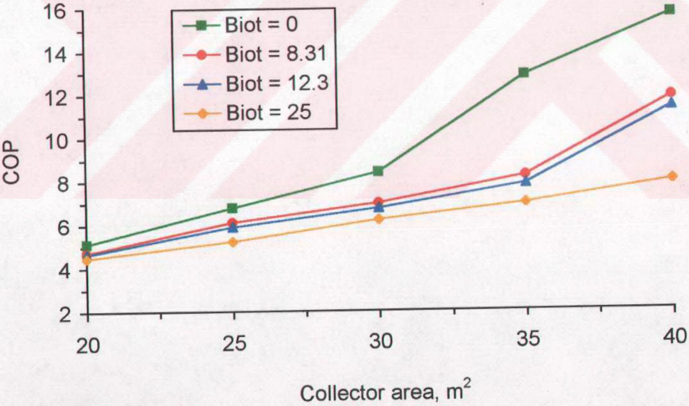


Figure 5.81. Annual coefficient of performance of the heat pump versus collector area for four different Biot numbers (Gaziantep, 100 house, one glass-cover black surface, β = 37.1 °, R = 25 m, Limestone)

Effect of earth type on annual energy fractions for two different Biot number of zero and 8.31 is depicted in Figures 5.82 and 5.83. These two figures show that the smallest heat pump work fraction and lost from the storage to the ground are obtained when the storage is surrounded with sand. The highest heat pump work fraction and the lowest lost, load, and solar energy fractions are obtained when the storage is embedded in granite. It is seen that there is a small effect of earth type on annual energy fractions, while there is a large effect of Biot number.

Figure 5.84 shows effect of Biot number on annual energy fractions. While the smallest Biot number gives the smallest heat pump work and lost fractions, it gives the highest solar energy and load fractions. But when the Biot number is increased, heat pump work and lost fractions increase, and solar energy and load fractions decrease. Lost fraction rises more rapidly than the heat pump work fractions and load fraction decreases more rapidly than the solar energy fraction.

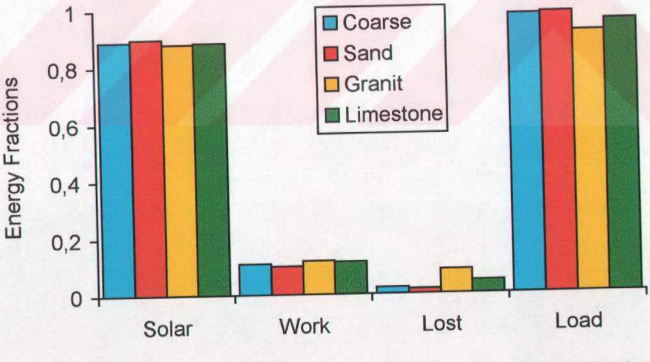


Figure 5.82. Effect of type of earth on annual energy fractions (Gaziantep, 100 houses, one glass-cover black surface, $A_c = 30\ m^2$, $\beta = 37.1\ ^\circ$, R = 25 m, Biot=0)

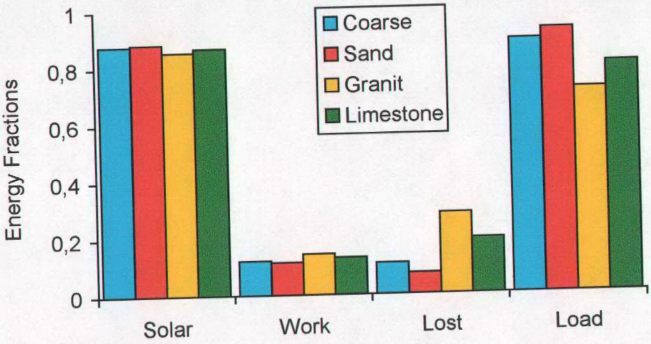


Figure 5.83. Effect of type of earth on annual energy fractions (Gaziantep, 100 houses, one glass-cover black surface, $A_c = 30$ m^2, $\beta = 37.1$ °, R = 25 m, Biot=25)

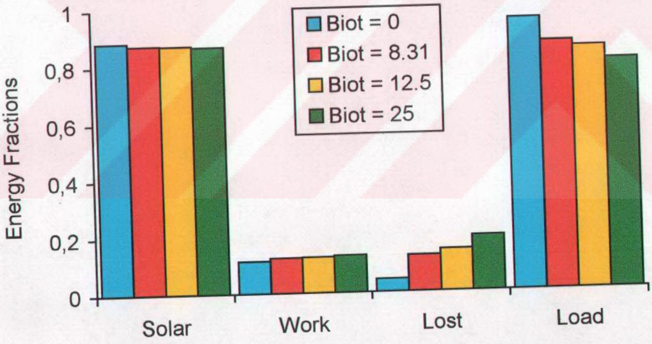


Figure 5.84. Effect of Biot number on annual energy fractions (Gaziantep, 100 houses, one glass-cover black surface, $A_c = 30$ m^2, $\beta = 37.1$ °, R = 25 m, Limestone)

Figure 5.85 denotes comparisons of annual water temperatures for two systems with spherical and hemispherical storages with the same size. It is seen from the Figure that annual storage temperature in two storages are equal each other. The same temperature gives same system performance parameters for the both systems.

Figures 5.86-5.89 show the effect of collector slope on annual solar fraction for comparison of results of İnallı et al.[36] and present study for the same conditions. It can be seen that the behavior of the figures are similar to each other. But there is a difference between the results which range 2-5 percents. There are several reasons for these differences in the results. These are: convergence factor, number of elements of series, numbers of eigenvalues, and many other input parameters for the simulation program. It is observed that the results of two studies agree with each other.

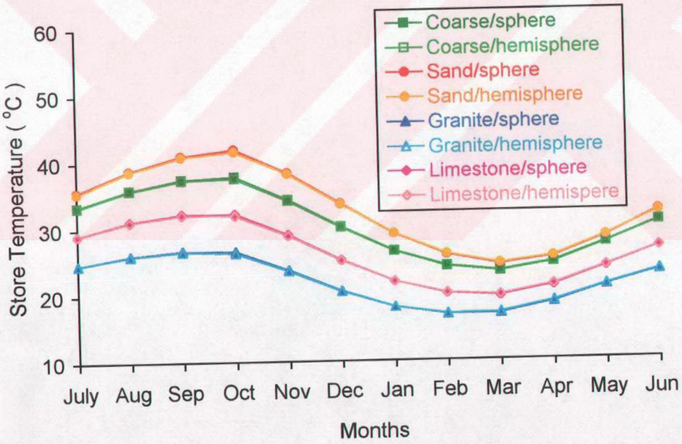


Figure 5.85. Comparison of annual water temperature for two heating systems with a spherical and hemispherical storages (Gaziantep, one house, one glass-cover black surface, A_c = 30 m^2, β = 37.1 °, R = a = 5 m, Biot=0.0, θ_0=180)

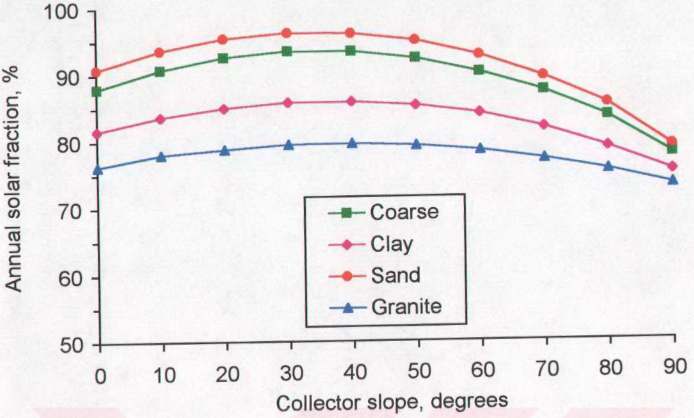


Figure 5.86- Annual solar fraction versus collector slope for different earth types (Elazığ, one house, two glass-cover black surface, $A_c = 40\ m^2$, İnallı et. al.[36])

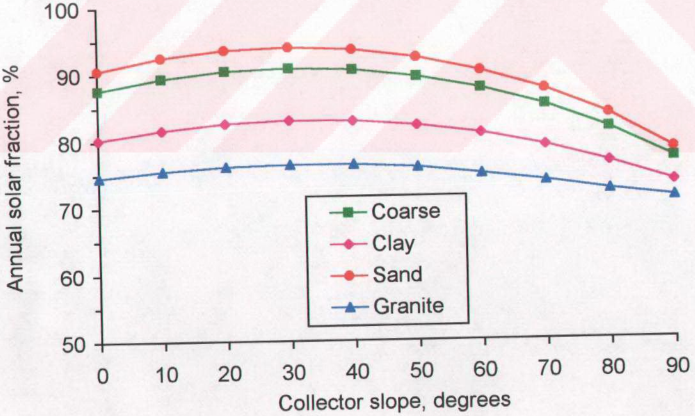


Figure 5.87. Annual solar fraction versus collector slope for different earth types (Elazığ, one house, two glass-cover black surface, $A_c = 40\ m^2$, Biot= 0)

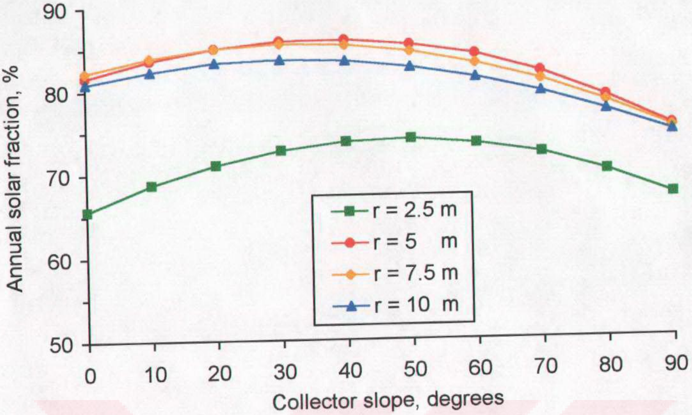


Figure 5.88. Annual solar fraction versus collector slope for different storage sizes (Elazığ, one house, two glass-cover black surface, $A_c = 40\ m^2$, İnallı et. al.[36])

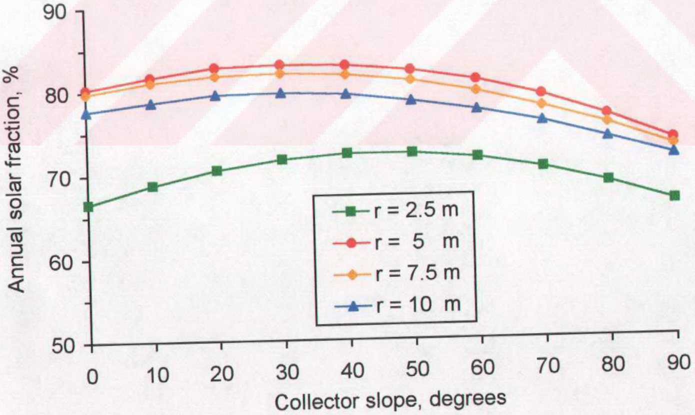


Figure 5.89. Annual solar fraction versus collector slope for different storage sizes (Elazığ, one house, two glass-cover black surface, $A_c = 40\ m^2$, Biot= 0)

CHAPTER VI

CONCLUSIONS AND RECOMMENDATIONS

6.1 CONCLUSIONS

In this thesis, annual periodic performances of solar aided heat pump systems with seasonal spherical and hemispherical underground thermal energy storages is investigated using analytical and computational methods. Effects of geological structure surrounding the storage, collector type, collector angle, collector surface area, storage size on the performance of the heating system are depicted and discussed. The followings can be concluded.

1. The highest water temperature in the storage is obtained at the end of the summer months of September and November, and the lowest storage temperatures occur at the end of the winter months of March and April.

2. The highest storage temperature and lowest collector efficiency occur when the storage is embedded in sand. The lowest water temperature in the storage and highest collector efficiency take place when the storage is buried in granite.

3. Effect of collector types on the storage temperature is quite low. Black paint-1 glass, selective surface-1 glass, and selective surface-2 glass give approximately same storage temperature while black paint-2 glass type collector gives 1-2 °C lower storage temperature. So, other performance parameters based on the storage temperature would be low.

4. The highest annual storage temperature, solar fraction, and mean water temperature are obtained as the collector slope is taken equal to the latitude angle for any region ($\beta = \phi_L$)

5. Amplitude of annual storage temperature increases when the storage decreases and system size increases.

6. Meteorological parameters have strong effects on the storage temperature. Meteorological data used in calculations for Gaziantep gives the highest storage temperature while data for İstanbul gives the lowest storage temperature. The highest heat lost and the lowest house heat load fraction are obtained for Gaziantep, while the lowest lost and the highest load fractions obtained for İstanbul.

7. Earth type has a strong effect on the storage temperature, collector efficiency, solar fraction, and heat pump COP for a small system with one house. Effect of earth type decreases when the system size is increased.

8. Storage size has a small effect on the annual solar fraction, heat pump COP, and annual energy fractions. But it has strong effect on the storage temperature and collector efficiency.

9. Larger system sizes and smaller Biot numbers give higher system performance.

10. For smaller Biot numbers, with a system for one house a storage volume of 250 m^3/house is suitable while with a system for 100 houses 200 m^3/house storage volume is required. On the other hand, for larger Biot numbers with a system for 100 houses a storage volume of 300 m^3/house can be used

11. It is concluded that annual water temperature in the spherical and hemispherical storages are equal to each other when the following assumptions are taken as: θ_0=180° and U_T=0.0

6. 2 RECOMMENDATIONS FOR FURTHER STUDY

The major objective of this study was the determination of the long term annual performance of solar aided space heating system utilizing a heat pump with spherical and hemispherical seasonal underground thermal energy storages. The system investigated was employed solar energy collection and dumping into seasonal underground storages throughout whole year. The amount of solar energy collection was estimated using the $\bar{\phi}$ method. The value of $\bar{\phi}$ for a monthly period depends upon the distribution of daily total solar radiation. The monthly average values of radiation on a horizontal surface, outside air, collector inlet temperatures, and transmittance absorptance product were used in the $\bar{\phi}$ method. However this study would become reality when the all of these values were used as hourly basis in calculations.

The following recommendations omitted in this study are given for further study:

1. Soil is assumed to be constant thermal properties and homogeneous structures in this investigation. The properties are dependent on the soil composition and moisture content. Energy transport in the soil occur by means of both conduction and convection. Convection heat transfer and phase change of moisture in the soil may be also involved in a further study.

2. Solar heat flux on free surface and convection heat transfer were not accounted in this study. Solar radiation incident on the free soil surface

around the storage and convection heat transfer may be taken into account using hourly solar heat flux and outside air temperature.

3. It may be investigated for isolated conditions of seasonal underground energy storages.

4. Economical analysis was not included in this study. This is quite a separate analysis which is at the moment out of scope of this work and can be the subject of another study.

5. The long term performance of the solar aided space heating system with a heat pump and seasonal energy storage should also be studied experimentally for several years of operation. After such an experimental study, one can test validity of the present study to find out how closely it predicts the results of an experimental study. Then the study will become comprehensive.

LIST OF REFERENCES

1. Wirth, P. E., 1955. “History of the development of heat pumps” Schweiz Bauzeigt, Vol. 73 (4), pp. 647-651.

2. Kemler, E., N. and Golesby, S., 1950. " Heat Pump Applications”, McGraw-Hill.

3. Crandall, A. C., 1946. “House heating with earth heat pump”, Electrical World, Vol. 126, No. 19, pp. 94-95

4. Ingersoll and Plass, 1948. "Theory of Ground Pipe Heat Source for The Heat Pump", Heating, Piping, and Air Conditioning, (July).

5. Hadley, W., A., 1949. "Operating Characteristics of Heat Pump Coils a Rational Correlation", New York, Edison Electric Ins.

6. Hooper, F., C., 1952. "An Experimental Residential Heat Pump", Canadian Journal of Technology, 30, 180-197.

7. Coogan, C., H., 1949. " Summary of Heat Absorption Rates for an Experimental ground coil system" Eng. Exp. Station Buil. No. 4 University of Connecticut, March.

8. Smith, G., S. 1951. "Factors Useful in Ground Grid Design for Heat Pumps", ASHVE Transactions, Vol. 57, 189.

9. Penrod, E., B., Knight, R., B. and Baker, M. 1950 "Earth Heat Pump Research, Part II", Eng. Exp. Station Bull., 5, (18), December, University of Kentucky.

10. Kidder, A., H. and Neher, J., H. 1952. . "Residential Heat Pump Experiments in Philadelphia. Earth as a Heat Source", AIEE Transactions, Part II, Application and Industry, Vol. 71, 343-350.

11. Harlow, J., H., and Klapper, C., E., 1952. "Residential Heat Pump Experiments in Philadelphia, Installation and Operating Experience, AIEE Transactions, Part II, Application and Industry, 71, 366-375.

12. Penrod, E., G., 1954. “Sizing earth heat pumps”, Refrigerating engineering, Vol. 62, No. 4, pp. 57-61.

13. Vestal, D., M., and Fluker, B., J., 1956. "Earth as Heat Source and Sink for Heat Pumps", Heating, Piping, and Air Conditioning, Vol. 28, (8), 117-123.

14. Smith, C., A., 1956. “Interim heat ground grids for heat pumps”, Heating, piping and air conditioning, Vol. 18, No. 8, pp. 127-133.

15. Freund, E. A. And Whitlow, G. S., 1959. “Earth Source Heat Pumps”, Transactions Am. Inst. Electrical Engineers, Vol. 77, pp. 540-551.

16. Griffiths, M., V., 1952. " Heat Transfer From Soil to Buried Pipes", ERA Report Y/T 18, Leatherhead, UK, Electrical Research Association.

17. Sumner, J., A., 1955. "Domestic Heating by The Heat Pump", Journal of Ins.Heating and Ventilating Engineers, pp. 129-151.

18. Cube, V. and Stiemle, F., 1978. Warmepumpen, Grundlagen und Praxis, Dusseldorf, VDI-Verlag GmbH.

19. Breger, D. S., Bankston, C. A., and Sunderland, J. E., 1992. "Central solar heating plants with seasonal storage", ASHRAE Journal, Vol. 34, No. 5, pp. 27-33.

20. Dalenback, J. O., 1990. "Central Solar Heating Plants with Seasonal Storage-Status Report", IEA Solar Heating and Cooling Program, Task VII. D14:1990. Stockholm, Sweden: Swedish Council for Building Research.

21. Ball, D. A., Fisher, R. D., and Hodgett, D. L., 1983, "Design Methods for Ground-Source Heat Pumps", ASHRAE Transactions, Part 2B, pp. 416-440.

22. Kavanaugh, S., 1992. "Ground-coupled heat pumps for commercial buildings", ASHRAE Journal, pp. 30-37.

23. Geeraert, B. and Steffens, J. C., 1979. "The use of soil with and without artificial heat generation as a heat source for the heat pump", Preceedings of new ways to save energy, Commission of European Communities, Brussels.

24. Taşdemiroğlu E., 1988. "Solar Energy Utilization Technical and Economical Aspects", Middle East Technical University, Ankara.

25. Lund P. D., 1984. "Optimisation of a community solar heating system with a heat pump and seasonal storage", Solar Energy, Vol. 33, No. 3/4, pp. 353-361.

26. Reid, R. L., McGraw, B. A. and Bedinger, A. F. G., 1985. "Performance of series solar assisted heat pump system in TECH House I", Journal of Solar Energy Engineering, Vol. 107, pp. 293-296.

27. Johnson, W. S., McGraw, B. A., Conlin, F., Wix, S. D., and Baugh, R. N., 1986. "Annual performance of a horizontal coil ground-coupled heat pump", ASHRAE Transactions, Part I, pp. 173-185.

28. Lund P. D., 1986. "Computational simulation of district solar heating systems with seasonal thermal energy storage", Solar Energy, Vol. 36, No. 5, pp. 397-408.

29. Lei, T. K., 1986. "Development of A Computational Model for A Ground-Coupled Heat Exchanger", ASHRAE Transactions, pp. 149-159.

30. Martin, S. D., 1990. "A design and economic sensitivity study of single-pipe horizontal ground-coupled heat pump systems", ASHRAE Transactions, Part I, pp.634-642.

31. Ünsal, M., 1990. "Recent Advances in Fourier Analysis and its applications", Edited by J. S. Byrnes and J. F. Byrnes, Kluwer Academic Publishers, Dordrecth, The Netharlands, pp. 59-72.

32. Inalli, M., 1993. "Toprak altında depolanan güneş enerjisi ile beslenen ısı pompalı bir konut ısıtma sisteminin bilgisayarda simülasyonu", Doktora Tezi, Fırat Üniversitesi, Fen Bilimleri Enstitüsü, Elazığ.

33. Tarnawski, S. D., Leong, W. H., 1989. "Pipe Spacing for Double-Layer Ground Heat Exchangers" ASHRAE Transactions, Part 2, 209-214.

34. Lund, P. D., and Östman, M. B., 1985. "A Numerical Model For Seasonal Storage of Solar Heat in The Ground by Vertical Pipes", Solar Energy, Vol. 34, No. 4/5, pp. 351-366.

35. Brunström, C., Larsson, M., Holst, P., Zinko, H., Hillström, C-G., 1985. "The Lyckebo Rock Cavern Seasonal Storage Plant After One Year of Operation", Sun World, Vol. 9, No. 3, pp. 93-95.

36. İnallı, M., Ünsal, M., and Tanyıldızı, V., 1997. "A Computational Model of A Domestic Solar Heating System with Underground Spherical Thermal Storage", Energy, Vol. 22, No. 12, pp.1163-1172.

37. İnallı, M. and Ünsal, M., 1997. "Yeraltında Mevsimlik Depolanan Güneş Enerjisi ile Beslenen Isı Pompalı Bir Konut Isıtma Sisteminin Analizi", Tr. Journal of Engineering and Environmental Sciences, 21, pp. 363-370.

38. Hahne, E. and Hornberger, M., 1994. "Experience with A Solar Heating ATES System for A University Building", Journal of Solar Energy Engineering, Vol. 116, pp. 88-93.

39. Shelton, J., 1975. "Underground Storage of Heat in Solar Heating Systems", Solar Energy, Vol. 17, pp. 137-143.

40. Ayhan, T. And Çomaklı, Ö. K., 1993. "Solar-assisted heat pump and energy storage for residential heating", Solar Energy, Vol. 51, No. 5, pp. 357-366.

41. Catan and Baxter, 1986. "An optimized Ground coupled Heat Pump System design for northern climate applications', ASHRAE Transactions,

42. Metz, D., 1983. "Ground-Coupled Heat Pump System Experimental Results", ASHRAE Transactions, Vol. 89, Part 2B, pp. 407-415.

43. Mohammad-zadeh, Johnson, R. R., Edwards, J. A., Safemazandarani, P., 1989. "Model Validation for Ground-Coupled Heat Pumps", ASHRAE Transactions, Part 2, pp. 215-221.

44. Mei, V. C. and Baxter, V. D., 1986. "Performance of a Ground-Coupled Heat Pump with Multiple Dissimilar U-tube Coils in Series", ASHRAE Transactions, Part 2A, pp. 30-41.

45. Phettaplace, G., Ueda, H. and Carbee, D., 1992. "Performance of Ground-Coupled Heat Pumps in Military Housing Units", Solar Engineering, Vol. 1, pp. 377-383.

46. Mei, V. C., and Fisher, S. K. 1983. "Vertical concentric tube ground coupled heat exchangers", ASHRAE Transactions, Vol. 89, Part 2, pp. 391-406.

47. Edwards, J. A., Sefamazandarani, P., Johnson, R. R., and Mohammadzadeh, Y., 1988. "Performance Test of A Direct Expansion Heat Pump System", ASHRAE Transactions, Part I, pp.318-329.

48. Kavanaugh, Steve P., 1992. "Simulation of Ground-Coupled Heat Pumps with an Analytical Solution", Solar Engineering, Vol. 1, pp. 395-409.

49. Johnson, W. S., Baugh, R. N., Griffith, W. A. and McGraw, B. A. 1987. "Seasonal Performance Evaluation of Two Horizontal-Coil Ground-Coupled Heat Pump Systems", ASHRAE Transactions, Part 2, pp. 1875-1885,

50. Beck, J. V., MacLain, H. A., Karnitz, M. A., Shonder, J. A., and Segan, E. G. 1988. "Heat Losses from Underground Steam Pipelines", Transactions of the ASME, Vol. 110, pp. 814-820.

51. Kavanaugh, S. P., 1992. "Simulation of Ground-Coupled Heat Pumps with an Analytical Solution", Solar Engineering, Vol. 1, pp. 395-400.

52. Mei, V. C., 1988. "Heat Pump Ground Coil Analysis With Thermal Interference", Journal of Solar Energy Engineering, Vol. 110, pp. 67-73.

53. Calm, J. M., 1982. "Trends and advances in electric heat pump technologies", Proceedings of the sixth heat pump technology conference, Tulsa, Oklahoma.

54. Klein, S. A., 1978. "Calculation of Flat Plate Collector Utilizability", Solar Energy, Vol. 21, pp. 393-402.

55. Mitchell, J. C., Theliacker, J. C., and Klein, S. A., 1981. "Calculation of monthly average collector operating time and parasitic energy requirements", Solar Energy, Vol. 26, pp. 555.

56. Collares-Pereira, M. and Rabl, A., 1979b "The Average Distribution of Solar Radiation Correlations Between Diffuse and Hemispherical and Between daily and Hourly insulation values, Solar Energy, vol. 22(2), p. 155.

57. Lui, B. Y. H. and Jordan, R. C.,1977, "Availability of Solar Energy for Flat-plate Solar Heat Collectors", Chap. V, Applications of Solar Energy for Heating and Cooling of Buildings, ASHRAE GRP 170.

58. Whillier, A., 1953. "Solar energy collection and its utilization for house heating", Sc. D. thesis, MIT, Cambridge, Mass.,

59. Lui, B. Y. H. and Jordan, R. C., 1953. "A rational procedure for predicting the long term average performance of flat plate solar energy collectors", Solar Energy, Vol. 7, pp. 53.

60. Klein, S. A. and Backman, W. A., 1979. "A general design method for closed loop solar energy systems", Solar Energy, Vol. 22, pp. 269.

61. Lui, B. Y. H. and Jordan, R. C., 1960. “The Interrelationships and characteristic distribution of direct, diffuse and total radiation”, Solar Energy, Vol. 4, No. 3, pp. 1-19.

62. Duffie, J.A., and Beckman, W.A. Solar Energy Thermal Process, 8 Wiley Interscience, New York. 1974.

63. Simon, F. F., 1976. "Flat-plate Solar-collector Performance Evaluation with A solar Simulator As A Basis for Collector Selection and Performance Prediction", Solar Energy, Vol. 18 , 451-466.

64. Klein S. A., 1979, “Calculation of the Monthly Average Transmittance-absorbtance Product”, Solar Energy, Vol. 23, pp. 547-551.

65. Dağsöz, A. K., 1995. “Türkiye’de Derece-Gün Sayıları Ulusal Enerji Tasarrufu Politikası Yapılarda Isı Yalıtımı”, İ. T. Ü. Makina Fakültesi, Isı Geçişi ve Ekonomisi Birimi.

66. Ünsal, M., Doğantan Z. S., 1981. "Güneş Enerjisi Takviyeli Isıtma Sistemleri İçin Aylık Ortalama Geçiş-yutma Sayısı", Isı Bilimi ve Tekniği Dergisi, 4 (1), 35-44.

67. Özışık, M. N., 1985. “Heat Transfer”, McGraw-Hill Book Company, Singapore.

68. Kakaç, S. and Yaman Y., 1979. “Heat Conduction”, Middle East Technical University, Publication Number: 60, Ankara.

69. Braun, J. E., Klein, S. A. And Mitchell, J. W., 1981. “Seasonal Storage of Energy in Solar Heating”, Solar Energy, Vol. 26, pp. 403-411.

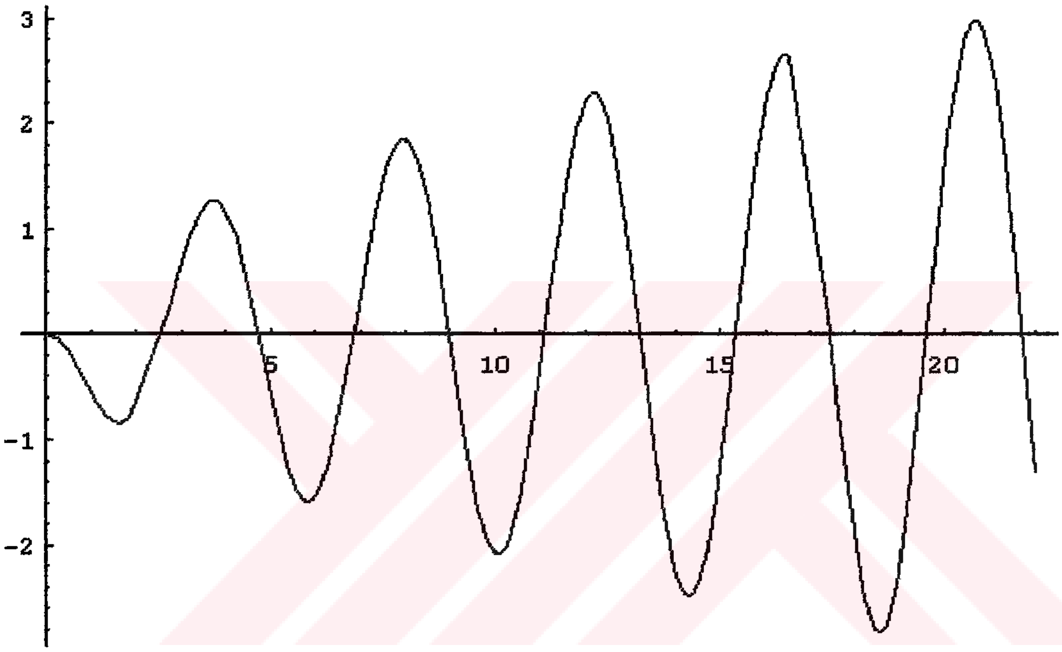
APPENDIX A

LISTING OF PROGRAM IN MATHEMATICA

Solution of Eigenvalue Problem in θ-direction

For Biot = 0

```
Bix :=0.
tetao :=85.*N[Pi]/180.
ni:=1; nf:=10; mi:=1; mf:=10;
f[a_] := Bix*BesselJ[0,a*tetao]-a*BesselJ[1,a*tetao]
Plot[f[a], {a,0.,22.} ]
```

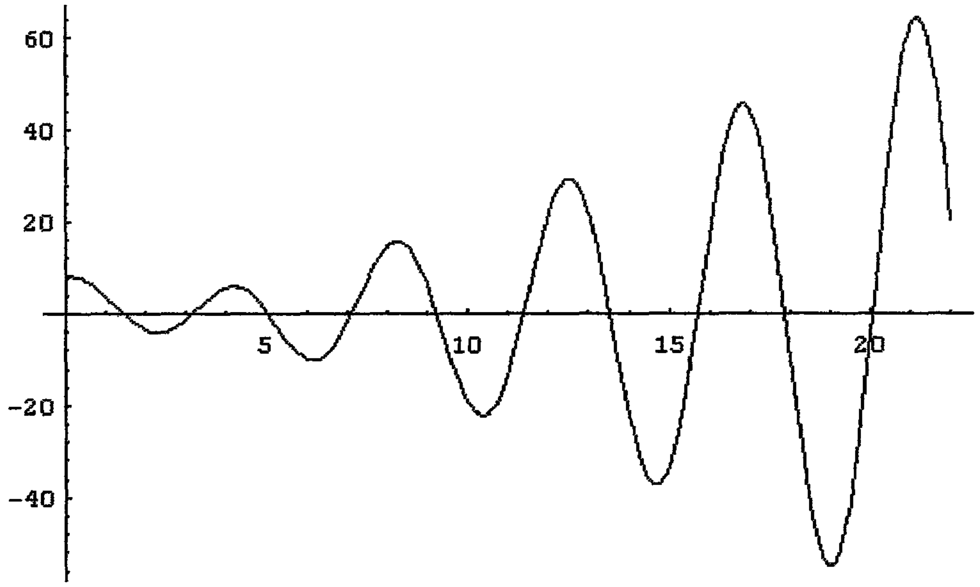


```
a[1]:=x/.FindRoot[f[x] == 0, {x,0.0 } ]
a[2]:=x/.FindRoot[f[x] == 0, {x,2.5 } ]
a[3]:=x/.FindRoot[f[x] == 0, {x,4.7 } ]
a[4]:=x/.FindRoot[f[x] == 0, {x,6.9 } ]
a[5]:=x/.FindRoot[f[x] == 0, {x,10.2 } ]
a[6]:=x/.FindRoot[f[x] == 0, {x,13.1 } ]
a[7]:=x/.FindRoot[f[x] == 0, {x,15.3 } ]
a[8]:=x/.FindRoot[f[x] == 0, {x,17.5 } ]
a[9]:=x/.FindRoot[f[x] == 0, {x,19.5 } ]
a[10]:=x/.FindRoot[f[x] == 0, {x,21.5 } ]
Do[Print["a[",n,"]:=",a[n]],{n,ni,nf}]
mu[1]:=Sqrt[0.25+a[1]^2]
mu1[1]:=mu[1]+1
Do[mu[n]=Sqrt[0.25+a[n]^2],{n,ni,nf}]
Do[mu1[n]=mu[n]+1,{n,ni,nf}]
fi[1]:=Sqrt[2.]/tetao
fipr[1]:=0.
```

```
Do[fi[m]=Sqrt[2.]*a[m]/(tetao*Sqrt[a[m]^2+Bix^2]*BesselJ[0,a[m]*tetao]),{m,2,
mf}]
Do[fipr[n]=-
Sqrt[2.]*a[n]^2./(tetao*Sqrt[a[n]^2+Bix^2]*BesselJ[0,a[n]*tetao]),{n,2,nf}]
Do[Do[bint[n,m]=0.0,{n,ni,ni}],{m,mi,mf}]
Do[Do[bint[n,m]=NIntegrate[(tet*Cot[tet]-
1.)*BesselJ[0,a[m]*tet]*BesselJ[1,a[n]*tet],
{tet,0.0,tetao}],{n,2,nf}],{m,mi,mf}]
Do[Do[bnm[n,m]=bint[n,m]*fipr[m]*fi[n],{n,ni,nf}],{m,mi,mf}]
Do[Do[Print["bnm[",m,",",n,"]:=",bnm[n,m]],{n,ni,nf}],{m,mi,mf}]
M1[n_]=Table[If[i==j,a[i]^2.-bnm[i,j],
        bnm[j,i]],{i,ni,nf},{j,mi,mf}]

MatrixForm[M1[nf]]

ev=Eigenvalues[M1[nf]]

ln=Sqrt[ev]

Do[Print["lnd[",n,"]:=",ln[[nf+1-n]]],{n,ni,nf}]

Vec=Eigenvectors[M1[nf]]
```

For z = 3 m, Biot = 8.31 and R = 25 m

```
Bix :=8.31
tetao :=83.1*N[Pi]/180.
ni:=1; nf:=10; mi:=1; mf:=10;
f[a_] := Bix*BesselJ[0,a*tetao]-a^2*BesselJ[1,a*tetao]
Plot[f[a], {a,0.,22.} ]
```

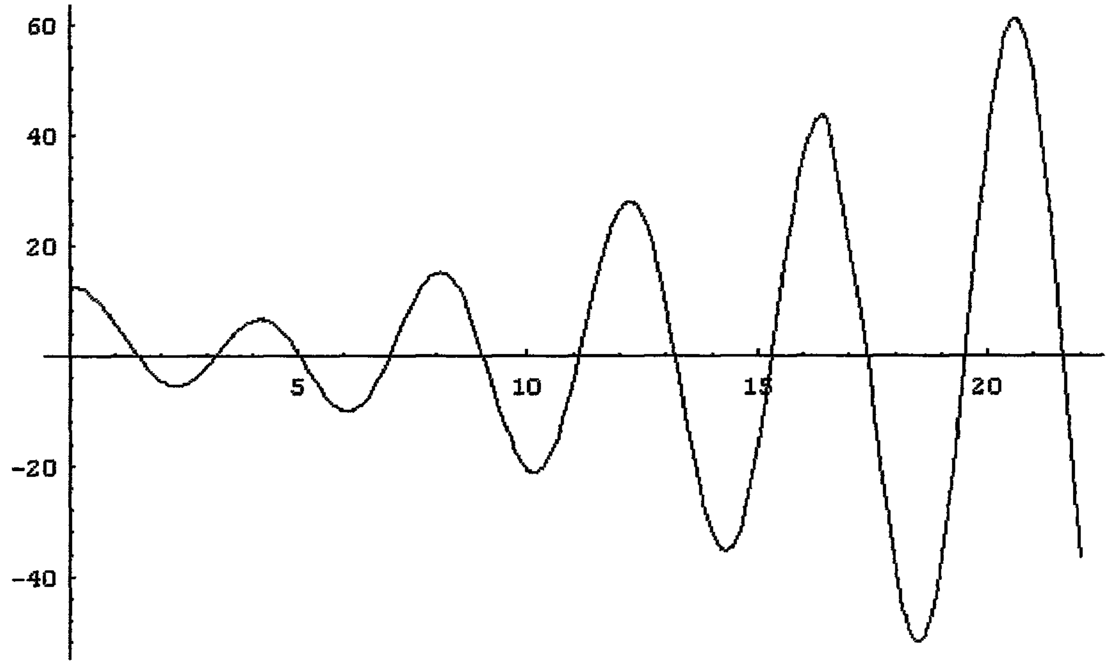


```
a[1]:=x/.FindRoot[f[x] == 0, {x,1.5 } ]
a[2]:=x/.FindRoot[f[x] == 0, {x,3.2 } ]
a[3]:=x/.FindRoot[f[x] == 0, {x,5. } ]
```

```
a[4]:=x/.FindRoot[f[x] == 0, {x,7. } ]
a[5]:=x/.FindRoot[f[x] == 0, {x,9 } ]
a[6]:=x/.FindRoot[f[x] == 0, {x,11 } ]
a[7]:=x/.FindRoot[f[x] == 0, {x,13 } ]
a[8]:=x/.FindRoot[f[x] == 0, {x,15 } ]
a[9]:=x/.FindRoot[f[x] == 0, {x,18 } ]
a[10]:=x/.FindRoot[f[x] == 0, {x,20 } ]
Do[Print["a[",n,"]:=",a[n]],{n,ni,nf}]
Do[Print[a[n]],{n,ni,nf}]
mu[1]:=Sqrt[0.25+a[1]^2]
mu1[1]:=mu[1]+1
Do[mu[n]=Sqrt[0.25+a[n]^2],{n,ni,nf}]
Do[mu1[n]=mu[n]+1,{n,ni,nf}]
fi[1]:=Sqrt[2.]/tetao
fipr[1]:=0.
Do[fi[m]=Sqrt[2.]*a[m]/(tetao*Sqrt[a[m]^2+Bix^2]*BesselJ[0,a[m]*tetao]),{m,2,
mf}]
Do[fipr[n]=-
Sqrt[2.]*a[n]^2./(tetao*Sqrt[a[n]^2+Bix^2]*BesselJ[0,a[n]*tetao]),{n,2,nf}]
Do[Do[bint[n,m]=0.0,{n,ni,ni}],{m,mi,mf}]
Do[Do[bint[n,m]=NIntegrate[(tet*Cot[tet]-
1.)*BesselJ[0,a[m]*tet]*BesselJ[1,a[n]*tet],
{tet,0.0,tetao}],{n,2,nf}],{m,mi,mf}]
Do[Do[bnm[n,m]=bint[n,m]*fipr[m]*fi[n],{n,ni,nf}],{m,mi,mf}]
Do[Do[Print["bnm[",m,",",n,"]:=",bnm[n,m]],{n,ni,nf}],{m,mi,mf}]
M1[n_]=Table[If[i==j,a[i]^2.-bnm[i,j],
        bnm[j,i]],{i,ni,nf},{j,mi,mf}]
MatrixForm[M1[nf]]
ev=Eigenvalues[M1[nf]]
ln=Sqrt[ev]
Do[Print["lnd[",n,"]:=",ln[[nf+1-n]]],{n,ni,nf}]
Vec=Eigenvectors[M1[nf]]
```

For z = 2 m, Biot = 12.5 and R = 25 m

```
Bix :=12.5
tetao :=85.41*N[Pi]/180.
ni:=1; nf:=10; mi:=1; mf:=10;
f[a_] := Bix*BesselJ[0,a*tetao]-a^2*BesselJ[1,a*tetao]
Plot[f[a], {a,0.,22.} ]
```

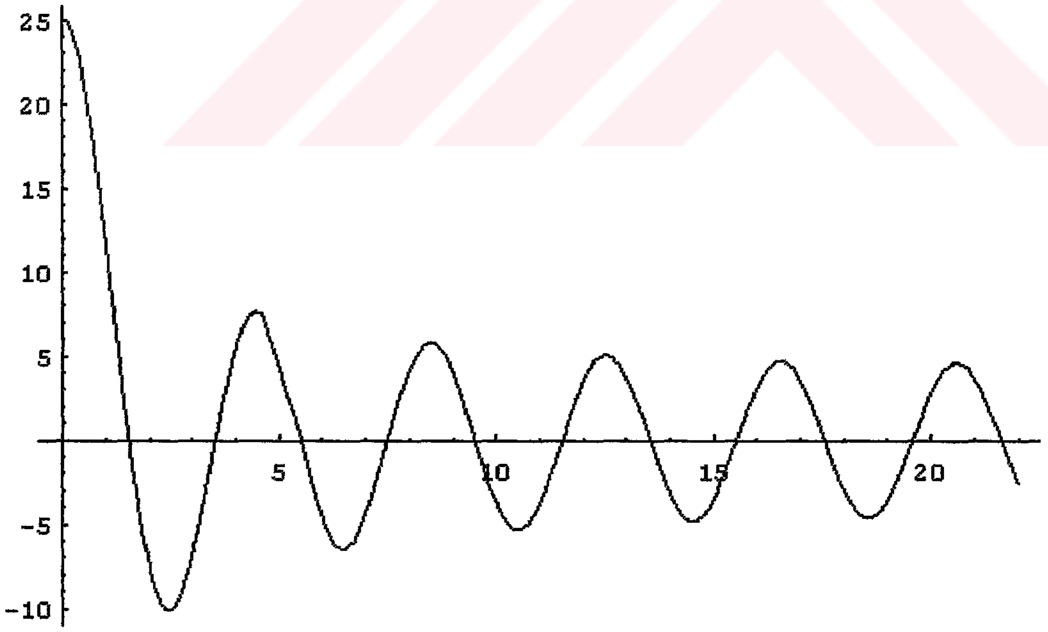


```
a[1]:=x/.FindRoot[f[x] == 0, {x,1.5 } ]
a[2]:=x/.FindRoot[f[x] == 0, {x,3.2 } ]
a[3]:=x/.FindRoot[f[x] == 0, {x,5. } ]
a[4]:=x/.FindRoot[f[x] == 0, {x,7. } ]
a[5]:=x/.FindRoot[f[x] == 0, {x,9 } ]
a[6]:=x/.FindRoot[f[x] == 0, {x,11 } ]
a[7]:=x/.FindRoot[f[x] == 0, {x,13 } ]
a[8]:=x/.FindRoot[f[x] == 0, {x,15 } ]
a[9]:=x/.FindRoot[f[x] == 0, {x,17 } ]
a[10]:=x/.FindRoot[f[x] == 0, {x,19 } ]
a[11]:=x/.FindRoot[f[x] == 0, {x,21 } ]
a[12]:=x/.FindRoot[f[x] == 0, {x,23 } ]
a[13]:=x/.FindRoot[f[x] == 0, {x,25. } ]
a[14]:=x/.FindRoot[f[x] == 0, {x,27. } ]
a[15]:=x/.FindRoot[f[x] == 0, {x,29 } ]
a[16]:=x/.FindRoot[f[x] == 0, {x,31 } ]
a[17]:=x/.FindRoot[f[x] == 0, {x,33.2 } ]
a[18]:=x/.FindRoot[f[x] == 0, {x,35.5 } ]
a[19]:=x/.FindRoot[f[x] == 0, {x,37.5 } ]
a[20]:=x/.FindRoot[f[x] == 0, {x,39.5 } ]

Do[Print["a[",n,"]:=",a[n]],{n,ni,nf}]
Do[Print[a[n]],{n,ni,nf}]

mu[1]:=Sqrt[0.25+a[1]^2]
mu1[1]:=mu[1]+1
Do[mu[n]=Sqrt[0.25+a[n]^2],{n,ni,nf}]
Do[mu1[n]=mu[n]+1,{n,ni,nf}]
fi[1]:=Sqrt[2.]/tetao
fipr[1]:=0.
```

```
Do[fi[m]=Sqrt[2.]*a[m]/(tetao*Sqrt[a[m]^2+Bix^2]*BesselJ[0,a[m]*tetao]),{m,2,
mf}]
Do[fipr[n]=-
Sqrt[2.]*a[n]^2./(tetao*Sqrt[a[n]^2+Bix^2]*BesselJ[0,a[n]*tetao]),{n,2,nf}]
Do[Do[bint[n,m]=0.0,{n,ni,ni}],{m,mi,mf}]
Do[Do[bint[n,m]=NIntegrate[(tet*Cot[tet]-
1.)*BesselJ[0,a[m]*tet]*BesselJ[1,a[n]*tet],
{tet,0.0,tetao}],{n,2,nf}],{m,mi,mf}]
Do[Do[bnm[n,m]=bint[n,m]*fipr[m]*fi[n],{n,ni,nf}],{m,mi,mf}]
Do[Do[Print["bnm[",m,",",n,"]:=",bnm[n,m]],{n,ni,nf}],{m,mi,mf}]
M1[n_]=Table[If[i==j,a[i]^2.-bnm[i,j],
        bnm[j,i]],{i,ni,nf},{j,mi,mf}]
MatrixForm[M1[nf]]
ev=Eigenvalues[M1[nf]]
In=Sqrt[ev]
Do[Print["Ind[",n,"]:=",In[[nf+1-n]]],{n,ni,nf}]
Vec=Eigenvectors[M1[nf]]
```

For z = 1 m, Biot = 25 and R = 25 m

```
Bix :=25.
tetao :=87.7*N[Pi]/180.
ni:=1; nf:=10; mi:=1; mf:=10;
f[a_] := Bix*BesselJ[0,a*tetao]-a*BesselJ[1,a*tetao]
Plot[f[a], {a,0.,22.} ]
```

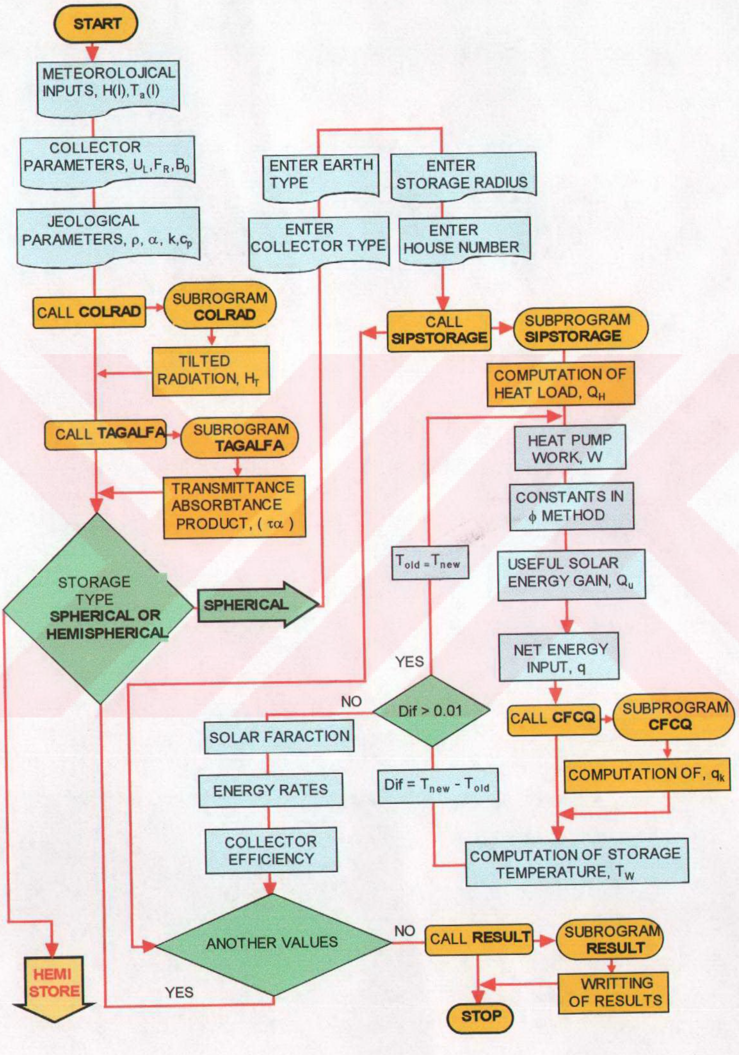


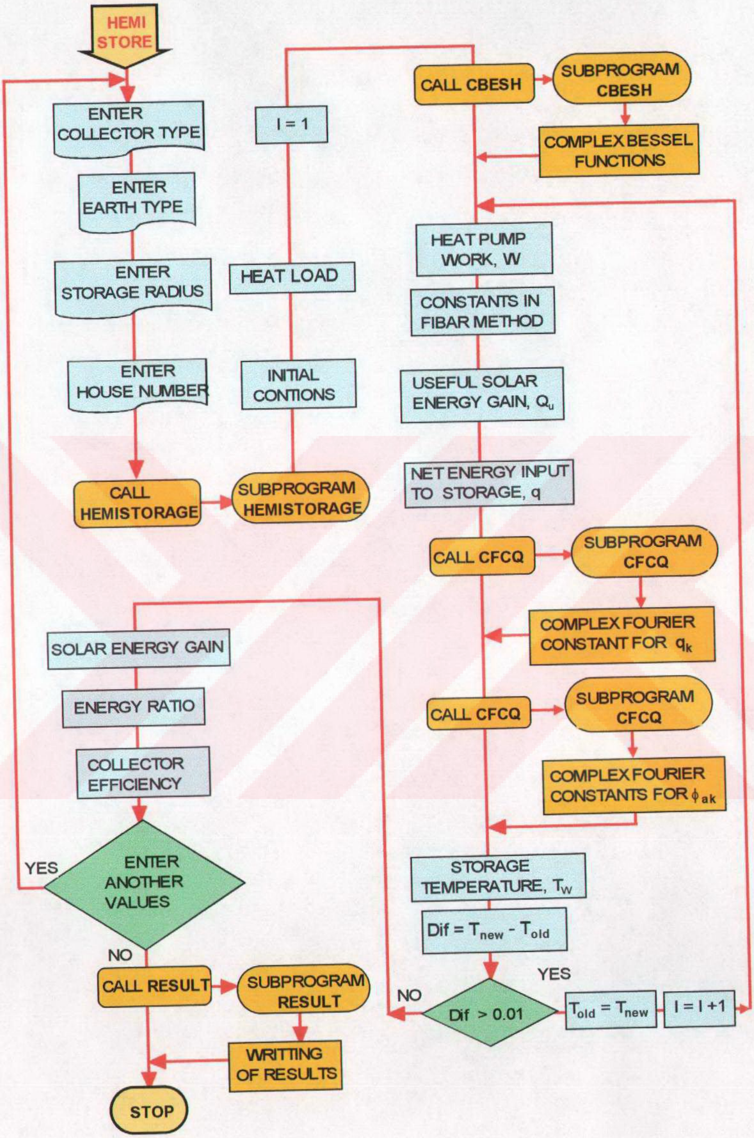
```
a[1]:=x/.FindRoot[f[x] == 0, {x,25.0 } ]
a[2]:=x/.FindRoot[f[x] == 0, {x,1.5 } ]
a[3]:=x/.FindRoot[f[x] == 0, {x,3.5 } ]
a[4]:=x/.FindRoot[f[x] == 0, {x,5.5 } ]
a[5]:=x/.FindRoot[f[x] == 0, {x,7.5 } ]
```

```
a[6]:=x/.FindRoot[f[x] == 0, {x,9.5 } ]
a[7]:=x/.FindRoot[f[x] == 0, {x,11.5 } ]
a[8]:=x/.FindRoot[f[x] == 0, {x,13.5 } ]
a[9]:=x/.FindRoot[f[x] == 0, {x,15.5 } ]
a[10]:=x/.FindRoot[f[x] == 0, {x,17.5 } ]

Do[Print["a[",n,"]:=",a[n]],{n,ni,nf}]
mu[1]:=Sqrt[0.25+a[1]^2]
mu1[1]:=mu[1]+1
Do[mu[n]=Sqrt[0.25+a[n]^2],{n,ni,nf}]
Do[mu1[n]=mu[n]+1,{n,ni,nf}]
fi[1]:=Sqrt[2.]/tetao
fipr[1]:=0.
Do[fi[m]=Sqrt[2.]*a[m]/(tetao*Sqrt[a[m]^2+Bix^2]*BesselJ[0,a[m]*tetao]),{m,2,
mf}]
Do[fipr[n]=-
Sqrt[2.]*a[n]^2./(tetao*Sqrt[a[n]^2+Bix^2]*BesselJ[0,a[n]*tetao]),{n,2,nf}]
Do[Do[bint[n,m]=0.0,{n,ni,ni}],{m,mi,mf}]
Do[Do[bint[n,m]=NIntegrate[(tet*Cot[tet]-
1.)*BesselJ[0,a[m]*tet]*BesselJ[1,a[n]*tet],
{tet,0.0,tetao}],{n,2,nf}],{m,mi,mf}]
Do[Do[bnm[n,m]=bint[n,m]*fipr[m]*fi[n],{n,ni,nf}],{m,mi,mf}]
Do[Do[Print["bnm[",m,",",n,"]:=",bnm[n,m]],{n,ni,nf}],{m,mi,mf}]
M1[n_]=Table[If[i==j,a[i]^2.-bnm[i,j],
        bnm[j,i]],{i,ni,nf},{j,mi,mf}]
MatrixForm[M1[nf]]
ev=Eigenvalues[M1[nf]]
In=Sqrt[ev]
Do[Print["Ind[",n,"]:=",In[[nf+1-n]]],{n,ni,nf}]
Vec=Eigenvectors[M1[nf]]
```

APPENDIX B

FLOWCHART FOR SIMULATION PROGRAM SOLARSTORE




HEMI STORE
ENTER COLLECTOR TYPE
ENTER EARTH TYPE
ENTER STORAGE RADIUS
ENTER HOUSE NUMBER
CALL HEMISTORAGE
SUBPROGRAM HEMISTORAGE
INITIAL CONTIONS
HEAT LOAD
I = 1
CALL CBESH
SUBPROGRAM CBESH
COMPLEX BESSEL FUNCTIONS
HEAT PUMP WORK, W
CONSTANTS IN FIBAR METHOD
USEFUL SOLAR ENERGY GAIN, Q_u
NET ENERGY INPUT TO STORAGE, q
CALL CFCQ
SUBPROGRAM CFCQ
COMPLEX FOURIER CONSTANT FOR q_k
CALL CFCQ
SUBPROGRAM CFCQ
COMPLEX FOURIER CONSTANTS FOR ϕ_{ak}
STORAGE TEMPERATURE, T_W
Dif = T_{new} - T_{old}
Dif > 0.01
YES
$T_{old} = T_{new}$
I = I + 1
NO
SOLAR ENERGY GAIN
ENERGY RATIO
COLLECTOR EFFICIENCY
ENTER ANOTHER VALUES
YES
NO
CALL RESULT
SUBPROGRAM RESULT
WRITTING OF RESULTS
STOP

```
C*************************************************************************************************
C*                                                                                               *
C* LISTING OF SIMULATION PROGRAM FOR THE HEATING SYSTEM                                          *
C*                                                                                               *
C*************************************************************************************************
                              PROGRAM SOLARSTORE
C*************************************************************************************************
   REAL QN(12),FBAR(12),IHM(5,12),XCB(12),TALF(12),NOH,Fb(5),Tb(5)
   REAL HH(12),DLTA(12),RBA(12),HT(12),RT(12),RN(12),HO(12),KTM(12)
  $,AA(12),BB(12),CC(12),RR(12),REF(12),Ta(5,12),Tair(60),Hrad(60),
  &Ef(12),Tnew,Ts1(12),Tx(12)
   CHARACTER*33,SL(5),TYP(10),City(5),Aylr(12)
   DATA REF/.2,.2,.2,.2,.2,.2,.3,.7,.5,.3,.2,.2/
   DATA Tair/23.1,23.3,18.4,12.9,7.7,2.5,0.3,1.,4.7,11.1,16.1,20.,
  E27.2,27.,22.,14.8,7.8,1.5,-1.3,0.,4.7,11.8,17.4,22.9,
  G27.1,26.9,22.2,15.3,9.4,4.5,2.6,3.8,7.2,12.7,18.2,23.7,
  I23.2,23.3,19.7,15.5,11.9,8.,5.1,5.5,6.7,10.9,15.8,20.6,
  i27.5,27.,22.7,18.3,13.9,10.4,8.2,8.8,11.1,15.3,20.1,24.7/
   Data Hrad/23.7,22.3,17.6,12.5,7.4,4.9,6.,8.5,12.7,17.9,20.9,22.1,
  E20.9,19.9,15.6,10.2,5.9,4.6,4.9,8.,10.9,15.3,17.2,21.1,
  G22.1,21.1,16.8,11.7,7.9,5.7,6.2,9.2,12.6,17.7,18.3,22.6,
  I21.5,19.8,14.9,10.,6.7,4.5,4.4,7.4,11.6,16.2,18.9,21.3,
  i19.,17.6,13.9,10.1,6.4,4.4,5.1,7.1,10.3,14.4,19.1,18.3/
   Data Aylr/'July','August','September','October','November',
  &'December','January','February','March','April','May','June'/
   DATA SL/'Coarse','Clay','Sand','Granite','Limestone'/
   DATA TYP/'1 Glass Cover Black Surface','2 Glass Cover
  &Black Surface','1 Glass Cover Selective Surface',
  &'2 Glass Cover Selective Surface'
  $,'Enter Relaxation Number','Enter Radius of Store',
  $'Enter Number of House','Enter Type of Soil','Enter Collector Area
  &','Enter Collector Slope'/
   Data City/'Ankara','Elazig','Gaziantep','Istanbul','izmir'/
   Data Fb/40.,38.7,37.1,40.,38.5/,Tb/40.,38.7,37.1,40.,38.5/
C***********************************************************************************************
   i1=0
   Do 23 i=1,5
   Do 22 j=1,12
   i1=i1+1
   Ta(i,j)=Tair(i1)
   Ihm(i,j)=Hrad(i1)
   If(j.Eq.12) go to 23
 22 Continue
 23 Continue
c   5 WRITE(6,33)City(1),City(2),City(3),City(4),City(5)
c  33 FORMAT(6(/),13X,50('*'),/13X,'*',48(' '),'*',/,
c    $13X,'*',4X,'If City is ',A9,' PRINT ---> 1    ',7X,'*',/,
c    $13X,'*',4X,'           ',A9,' PRINT ---> 2    ',7X,'*',/,
c    $13X,'*',4X,'           ',A9,' PRINT ---> 3    ',7X,'*',/,
c    $13X,'*',4X,'           ',A9,' PRINT ---> 4    ',7X,'*',/,
c    $13X,'*',4X,'           ',A9,' PRINT ---> 5    ',7X,'*',/,
c    $13X,'*',7X,'     NOT ANY ONE, PRINT ---> other',7X,'*',/,
c    $13X,'*',48(' '),'*',/,13X,50('*'))
c     READ(5,*)il
  5 il=3
   IF(il.GT.5)GO TO 12
   WRITE(6,7)TYP(9)
   READ(5,*) Ac
c     Ac=30
c     WRITE(6,7)TYP(10)
```

```
c     READ(5,*) Tb(il)
C***********************************************************************
      CALL COLRAD(il,Fb,Tb,HH,DLTA,RBA,HT,RT,RN,HO,KTM,AA,BB,CC,
     &RR,Ref,IHM)
C***********************************************************************
c      WRITE(6,67)Typ(1),Typ(2),Typ(3),Typ(4)
c  67 FORMAT(12(/),8X,61('*'),/8X,'*',23X,'OPTIONS',29X,'*',/,
c     $8X,'*',4X,'FOR ',A33,', PRINT —> 1 ',3X,'*',/,
c     $8X,'*',4X,'FOR ',A33,', PRINT —> 2 ',3X,'*',/,
c     $8X,'*',4X,'FOR ',A33,', PRINT —> 3 ',3X,'*',/,
c     $8X,'*',4X,'FOR ',A33,', PRINT —> 4 ',3X,'*',/,
c     $8X,'*',30X,'NOT ANY ONE, PRINT —> 5 ',3X,'*',/,
c     $8X,'*',59(' '),'*',/8X,61('*'),4(/))
c      READ(5,*)M
      M=1
      IF(M.EQ.5) GO TO 12
C***********************************************************************
      CALL TAGALFA(il,Fb,Tb,Ac,Ref,HH,DLTA,RBA,M,TALF,FR)
C***********************************************************************
      WRITE(6,7)Typ(5)
    7 FORMAT(15(/),20X,33('*'),/20X,'*',31(' '),'*',/20X,'*',4X,A23
     $,4X,'*',/20X,'*',31(' '),'*',/20X,33('*'),5(/))
C      READ(5,*)RL
C  Rl:Relaxiation Constant
      Rl=.01

c      WRITE(6,7)TYP(6)
c      READ(5,*)R
       R=25.
c      WRITE(6,7)TYP(7)
c      READ(5,*)NOH
      Noh=100

c      WRITE(6,7)TYP(8)
c      WRITE(6,11)sl(1),Sl(2),Sl(3),Sl(4),Sl(5)
c  11 FORMAT(13X,50('*'),/13X,'*',48(' '),'*',/,
c     $13X,'*',4X,'If Soil Is ',A9,' PRINT —> 1    ',7X,'*',/,
c     $13X,'*',4X,'           ',A9,' PRINT —> 2    ',7X,'*',/,
c     $13X,'*',4X,'           ',A9,' PRINT —> 3    ',7X,'*',/,
c     $13X,'*',4X,'           ',A9,' PRINT —> 4    ',7X,'*',/,
c     $13X,'*',4X,'           ',A9,' PRINT —> 5    ',7X,'*',/,
c     $13X,'*',9X,'    NOT ANY ONE, PRINT ---> other',6X,'*',/,
c     $13X,'*',48(' '),'*',/,13X,50('*')/)
c      READ(5,*)L
      L=5
      IF(L.GT.5)GO TO 12
C*****************************************************************************************************
      CALL HEMISTORE(il,L,M,Noh,HT,RT,RN,IHM,AA,BB,CC,RR,Ta,FR,XCB,Rl,
     &FBAR,Ac,R,Talf,Ef,Tnew,Ts1,Tx,aylr,QSY,WY,QLT,QLY,QY,R1,R2,R4,R5)
C*****************************************************************************************************
      CALL RESULT(il,M,L,Tb,Ac,R,NOH,Ta,Sl,Aylr,Typ,City,IHM
     $,KTM,RR,RN,XCB,FBAR,Ef,Tnew,Ts1,Tx,QSY,WY,QLT,QLY,QY,R1,R2,R4,R5)
C*****************************************************************************************************
c      WRITE(6,*)' IF YOU WANT TO CONTINUE ENETER ANY BUTTON'
c      READ(5,15)C
c  15 FORMAT(A5)
   12 WRITE(6,13)
   13 FORMAT(20(/),13X,46('*'),/13X,'*',44(' '),'*',/,
     $13X,'*',4X,'  WILL YOU ENTER ANOTHER VALUES  ',7X,'*',/,
     $13X,'*',4X,'        IF YES  ENTER —> 1  ',7X,'*',/,
     $13X,'*',4X,'        IF NOT  ENTER —> ANY ',7X,'*',/,
```

```
     $13X,'*',44(' '),'*'/,13X,46('*'),10(/))
      READ(5,*)K
c     K=3
c   12 K=1
      IF(K.EQ.1) GO TO 5
      STOP
      END
C*****************************************************************************************
C* THIS SUBROUTINE SUBPROGRAM CALCULATES MONTHLY AVERAGE PHERICAL
C* STORE TEMPERATURE BY ITERATION METHOD AND ENERGY FRACTIONS         *
C*****************************************************************************************
      SUBROUTINE HEMISTORE(il,L,M,Noh,HT,RT,RN,IHM,AA,BB,CC,RR,Ta,FR,XCB
     &,RI,FBAR,Ac,R,Talf,Ef,Tnew,Ts1,Tx,aylr,
     &QSY,WY,QLT,QLY,QY,R1,R2,R4,R5)
C*****************************************************************************************
      COMPLEX Smke,Smka,Smne,Smna,Pvc
      Complex S2,Say,Sayx,Sayx0,Sayxe,Sayxa,Fwke,Fwka,Exc
      COMPLEX QKA(20),FAKA(20),QKE(20),FAKE(20)
      Complex H1w(20,20),H2w(20,20),H1(20,20),H2(20,20),H2x(20,20)
      COMPLEX ze,za,Cy(1),We(20),Wa(20),Vke(20),Vka(20),H1x(20,20)
      Complex H1mu1,H1mu,H2mu1,H2mu,Sy1k,Sy1k0,Sy1ke,Sy1ka
      Real J0t,J0,J1,ln(10),btn(10),Mu(10),Kn(10)
      REAL RT(12),RN(12),IHM(5,12),FBAR(12),AA(12),BB(12),CC(12),HT(12)
      REAL RR(12),Rj(13),Pik(13),SBT(12),IMO(12),DTT(12),Fa(12),Ef(12)
      REAL H(12),Tag(12),WO(12),ITC(12),Xcb(12),Qum(12),Qn(12),S(12)
      REAL Fw(12),Ta(5,12),Tin(12),Told(12),Tnew(12),Ts1(12),Tx(12)
      REAL Noh,Talf(12),UL(4),alfa(5),Con(5),RO(5),CP(5),Tgrd(20,12)
      CHARACTER*33 AYLR(12)
C*********************************************************************
C  Biot = 0.0 Teta=179 i‡in ”zde§erler
c     Data Ln/0.,1.21967,2.23313,3.23832,4.24106,5.24276,6.24392,
c    &7.24476,8.24539,9.24589/
c     DATA Bi,tetdeg,Tetao,Utp/0.0,0.0,180.,.0/
C*********************************************************************
Cfor Biot=0.0 and Teta=85 i‡in ”zde§erler
      Data Ln/0.0,2.59101,4.73311,6.85689,13.2241,13.2308,15.3406,
     &17.4616,19.5798,21.6948/
      DATA Bi,tetdeg,Tetao,Utp/0.0,0.0,85.,.0/
C*********************************************************************
Cfor Ly=1 m, Biot=25.0 and Teta=87.7, R=25 meters
c     Data Ln/1.51047,3.3188,5.1036,6.96377,8.90623,10.8971,12.9132,
c    &14.9429,16.9802,19.0216/
c     Data Bi,tetdeg,Tetao,yt,hw/25.0,87.7,87.7,1.,500./
C*********************************************************************
Cfor Ly=2 m, Biot=12.5 and Teta=85.41, R=25 meters
c     Data Ln/1.4905,3.21123,5.02411,6.99452,9.04035,11.1169,13.207,
c    &15.304,17.4047,19.5076/
c     DATA Bi,tetdeg,Tetao,yt,hw/12.5,85.41,85.41,2.,500./
C*********************************************************************
Cfor L=3m, Bi=8.31-Teta0=83.1, R=25 m, Date:5.6.1998
c     Data Ln/1.47164,3.15701,5.05831,7.12709,9.25337,11.4003,
c    &13.5559,15.7158,17.878,20.0415/
c     DATA Bi,tetdeg,Tetao,yt,hw/8.31,0.,83.1,3.,500./
C*********************************************************************
C  Bi = 0.1 i‡in ”zde§erler
c     Data ln/0.34993,2.47514,4.48623,6.49046,8.49267,10.494 ,12.4949,
c    &14.4953,16.4952,18.4914/
C*********************************************************************
      DATA RO/2050.,1500.,1500.,2640.,2500./,CP/1.842,.88,.8,.82,.90/
      DATA ALFA/1.39E-7,1.1E-6,2.5E-7,1.4E-6,5.75E-7/,
     &Con/0.519,1.4,0.3,3.0,1.3/
```

```
      DATA SBT/0.,0.,0.,0.,1.,1.,1.,1.,1.,1.,0.,0./
      DATA IMO/31,31,30,31,30,31,31,28,31,30,31,30/,UL/7.4,4.5,5.,3.2/
      DATA Tinf,UA,FI,PI,U/288.,345.,0.01736,3.1415,1.2/
      DATA ni,nf,ke,ka,kf/1,10,-10,10,10/
              Open(unit=1,file='Flim-Bi0',status='replace')
C***************************************************************************
      ir=1
      delr=0.0
      X1=1.
      Teto=Pi*Tetao/180.
      P=(1-COS(Teto))*1000.*4.187/(3.*RO(L)*Cp(L))
      If(Bi.Eq.0.0) Go to 8
      Utp=1./(1/hw+yt/Con(L))
    8 Stp=Utp*R/(2.*Con(L))
      GAMA=2.*Pi*R*Con(L)/(NOH*UA)
      Hpk=NOH*UA/(2.*Pi*R*Con(L)*Tinf)
      Spk=NOH/(24.*3600.*2.*Pi*R*Con(L)*Tinf)
      Tp=ALFA(L)*31536000/(R*R)
      DO 10 J=1,13
      Rj(J)=(J-7)/12.
      Pik(J)=2*Pi*Rj(J)
   10 CONTINUE
      Fa0=0.0
      DO 44 I=1,12
      H(i)=Sbt(i)*Hpk*(20.-Ta(il,i))
      IF(H(i).LE.0.0) H(i)=0.0
      Tin(i)=15
      Told(I)=15
      Tag(I)=-1./2+I/12.
      DTT(I)=alfa(L)*3600.*24.*IMO(I)/(R*R)
      Fa(i)=(Ta(il,i)+273.)/Tinf-1.
      Fa0=Fa0+Fa(i)
   44 CONTINUE
      Fa0=Fa0/12.
      Fa0=0.
C***************************************************************************
          CALL FAIR(ke,ka,Pi,Pik,Fa,Fake,Faka)
C***************************************************************************
      Cos1=Cos(teto)-1.
      tet=tetdeg*Pi/180.
      DO 95 N=2,nf
      Y=ln(n)*tet
      J0t=BESJ0(Y)
      X=ln(n)*teto
      J0=BESJ0(X)
      J1=BESJ1(X)
      Eig=Sqrt(2.)/(teto*Sqrt(ln(n)**2.+Bi**2.))
      Kn(n)=eig*ln(n)*J0t/J0
      Btn(n)=eig*Sin(teto)*J1/J0
      write(6,91)n,Btn(n)
   91 Format(' Btn=',F6.3)
   95 CONTINUE
      Kn(1)=1./Sqrt(2.)
      Btn(1)=2.*Kn(1)
C*********************************************************************************
c Bessel fonksiyonlar□n□n hesaplanams□
C*********************************************************************************
      DO 4 k=1,kf
      Vk=Sqrt(Pi*k/tp)
      We(k)=Cmplx(-1.,-1.)*Vk
      Wa(k)=Cmplx(-1.,1.)*Vk
```

```
     Vke(k)=Cmplx(1,-1)*Vk
     Vka(k)=Cmplx(1,1)*Vk
   4 CONTINUE
1000 DO 6 n=1,nf
     mu(n)=Sqrt(0.25+ln(n)**2.)
     fnu1=mu(n)+1
     fnu=mu(n)
     DO 5 k=1,kf
     ze=We(k)
     CALL CBESH(ze,fnu1,1,2,1,Cy,Nz,Ierr)
     H2mu1=Cy(1)
     Call CBESH(ze,fnu,1,2,1,Cy,Nz,Ierr)
     H2mu=Cy(1)
     H2w(n,k)=We(k)*H2mu1/H2mu
     H1(n,k)=H2mu

     ze=We(k)*X1
     H1x(n,k)=Cy(1)
     za=Wa(k)
     H1mu1=Cy(1)
     H1mu=Cy(1)
     H1w(n,k)=Wa(k)*H1mu1/H1mu
     H2(n,k)=H1mu

     za=Wa(k)*X1
     H2x(n,k)=Cy(1)
   5 Continue
   6 Continue
C***********************************************************************************
     In=0
 500 In=In+1
     WRITE(6,251)In,AC,R,Noh
 251 FORMAT(70('*'),/' Itr.=',I3,' Ac=',F3.0,'  R=',F5.2,'  Noh=',F5.1)
     DO 60 I=1,12
     Fw(i)=(Tin(i)+273.)/Tinf-1.
     c1=U*(Fi-Fa(i))+Fi
     IF(Fw(i).GE.C1) Go to 58
     c2=c1-Fw(i)
     C3=(Tinf*(Fw(i)+1.)-173.)/70.
     C4=(308.-Tinf*(c1+1))/40.
     WO(i)=SBT(I)*(Fi-Fa(i))/(C3*ALOG((c1+1.)/c2)+C4)
     Go to 59
  58 WO(i)=0.0
  59 WO(I)=WO(I)/gama
  60 CONTINUE
C***************************************************************************************************************
C CALCULATION OF THE MONTHLY AVERAGE ENERGY INPUT TO THE STORE
C***************************************************************************************************************
     Q0=0.
     DO 70 I=1,12
     ITC(I)=UL(M)*(Tin(I)-Ta(il,I))/Talf(I)
     IF(ITC(I).LE.0.0)ITC(I)=0.0
     XCB(I)=ITC(I)/(RT(I)*RN(I)*Ihm(il,I))*0.0036
     FBAR(I)=EXP((AA(I)+BB(I)*RN(I)/RR(I))*(XCB(I)+CC(I)*XCB(I)**2))
     QUM(I)=Ac*Fr*Talf(I)*HT(I)*Fbar(I)*10**6.
     Ef(i)=Qum(i)*100./(Ac*Ht(i)*10**6.)
     S(I)=Spk*QUM(I)
     Qn(I)=S(I)-H(I)+WO(I)
     Q0=Q0+Qn(I)
  70 CONTINUE
     Q0=Q0/12.
```

```
C************************************************************************
       CALL QNET(ke,ka,Pi,Pik,Qn,Qke,Qka)
C************************************************************************
C              CALCULATION OF STORAGE TEMPERATURE
C************************************************************************
C      k = 0
c    --------
      Sumn=0.0
      Btsum=0.0
      Do 72 n=ni,nf
      Sumn=Sumn+Btn(n)**2.*(1.+2.*mu(n))/2.
      Btsum=Btsum+Btn(n)*Kn(n)
      Sayx0=Sayx0+X1**(0.5-mu(n))
   72 Continue
      Fw0=(Q0+(Stp+Sumn+Cos1)*Fa0)/(Stp+Sumn)
      Sy1k0=Btsum*(Fw0-Fa0)+Fa0
      Sayx0=Btsum*(Fw0-Fa0)+Fa0*(1.-X1)+Fa0
C*********************************************
      DO 400 I=1,12
c    --------
C      k < 0
c    --------
      Smke=CMPLX(0.,0.)
      Sy1ke=Cmplx(0.,0.)
      Sayxe=Cmplx(0.,0.)
      DO 201 K=-1,ke,-1
      k1=-1*k
      Smne=Cmplx(0.,0.)
      Sayx=Cmplx(0.,0.)
      DO 200 N=ni,nf
      Smne=Smne+Btn(n)**2.*((1.-2.*mu(n))/2.+H2w(n,k1))
      Sayx=Sayx+Btn(n)*Kn(n)*H1x(n,k1)/H1(n,k1)
  200 CONTINUE
      Pvc=Cmplx(0.,P*2.*Pi*k/tp)
      Fwke=(qke(k1)+(Stp+Smne+Cos1*(1.+Vke(k1)))*Fake(k1))
     &/(Pvc+Stp+Smne)
      Exc=Cmplx(COS(2*Pi*k*Tag(I)),Sin(2*Pi*k*Tag(I)))
      Smke=Smke+Fwke*Exc
      Sy1ke=Sy1ke+(Btsum*(Fwke-Fake(k1))+Fake(k1))*Exc
      S2=Fake(k1)*(Exp((1.-X1)*Vke(k1))-X1)
      Sayxe=Sayxe+((Sayx*(Fwke-Fake(k1))*(X1**0.5)+S2)/X1+
     &Fake(k1))*Exc
  201 CONTINUE
c    --------
C      k > 0
c    --------
      SMKA=CMPLX(0.,0.)
      Sy1ka=Cmplx(0.,0.)
      Sayxa=Cmplx(0.,0.)
      DO 301 K=1,ka
      Smna=CMPLX(0.,0.)
      Sayx=Cmplx(0.,0.)
      DO 300 N=ni,nf
      Smna=Smna+Btn(n)**2.*((1.-2.*mu(n))/2.+H1w(n,k))
      Sayx=Sayx+Btn(n)*Kn(n)*H2x(n,k)/H2(n,k)
  300 CONTINUE
      Pvc=CMPLX(0.,P*2.*Pi*k/tp)
      Fwka=(qka(k)+(Stp+Smna+Cos1*(1.+Vka(k)))*Faka(k))
     &/(Stp+Pvc+Smna)
      Exc=Cmplx(COS(2*Pi*k*Tag(I)),Sin(2*Pi*k*Tag(I)))
      Smka=Smka+Fwka*Exc
```

```
      Sy1ka=Sy1ka+(Btsum*(Fwka-Faka(k))+Faka(k))*Exc
      S2=Faka(k)*(Exp((1.-X1)*Vka(k))-X1)
      Sayxa=Sayxa+((Sayx*(Fwka-Faka(k))*(X1**.5)+S2)/X1+
     &Faka(k))*Exc
 301 CONTINUE

      Fw(i)=Fw0+Smke+Smka
      Tnew(i)=(Fw(i)+1)*Tinf-273.

      Sy1k=Sy1k0+Sy1ke+Sy1ka
      Ts1(i)=(Sy1k+1)*Tinf-273.

       Say=Sayx0+Sayxe+Sayxa
       Tx(i)=(Say+1)*Tinf-273.
      WRITE(6,188)AYLR(I),Tnew(I),Ts1(i),Tx(i)
 188 FORMAT(' Tstr(',A3,')= ',F7.2,'   Tsy= ',F6.2,'   Tsx= ',F6.2)
 400 CONTINUE
C*************************************************************************
      DO 12 I=1,12
      Frk=ABS(Tnew(I)-Told(I))
      IF(Frk.GT..01) GO TO 13
  12 CONTINUE
      GO TO 55
  13 DO 14 I=1,12
      Tin(I)=Tin(I)+RL*(Tnew(I)-Tin(I))
      Told(I)=Tnew(I)
  14 CONTINUE
      GO TO 500
  55 WRITE(6,351)x1,In,Bi,Tetdeg,DelR
      WRITE(1,351)x1,In,Bi,Tetdeg,DelR
 351 FORMAT(10('*')'   The Program APhd.For Results at X=',F5.2,
     &10('*'),/' Itr.=',I3,' Biot=',F5.2,'   Tetdeg=',F6.2,
     &/' Monthly Average Earth Temperature at  DelR=',F5.2)
      Do 125 i=1,12
      Tgrd(ir,i)=Tx(i)
      Write(1,121)Tx(i)
 121 Format(F6.2)
 125 Continue
c      Write(6,*)' Baska "DeltaR" ler girmek istermisiniz'
c      Write(6,*)' Evet ise Depodan Uzak Mesafeyi Giriniz'
c      Write(6,*)' Hay□r ise 22 giriniz                    '
c      Read(6,*)DelR
      DelR=delr+0.5
      X1=1.+DelR/R
      If(ir.Eq.15)go to 333
      ir=ir+1
      go to 1000
C*********************************************************************
 333 Do 340 i=1,10,3
      Write(1,*)' Monthly Average Earth Temperature For ',Aylr(i)
      Do 340 j=1,ir
      Write(1,337)Tgrd(j,i)
 337 Format(F6.2)
 340 Continue
      QY=0.0
      WY=0.0
      QSY=0.0
      QLY=0.0
      Tmn=0.0
      DO 216 I=1,12
      Tmn=Tmn+Tnew(i)
```

```
      QY=QY+QN(I)*DTT(I)
      QSY=QSY+S(I)*DTT(I)
      QLY=QLY+H(I)*DTT(I)
      WY=WY+WO(I)*DTT(I)
  216 CONTINUE
      QLT=QY
      Tmn=Tmn/12.
      F=(Qsy-Qlt)*100./Qly

      Write(1,*)'Monthly average Collector efficiency'
      Do 165 i=1,12
      Write(1,162)Ef(i)
  162 Format(F6.2)
  165 Continue

      Write(1,*)'Monthly Average Storage Water Temperature'
      Do 170 i=1,12
      Write(1,168)Tnew(i)
  168 Format(F6.2)
  170 Continue

      RTOT=QSY+WY
      R1=QSY/RTOT
      R2=WY/RTOT
      R4=QLT/RTOT
      R5=QLY/RTOT
      Write(1,*)'Yearly Energy Fractions'
      Write(1,558)R1,R2,R4,R5
  558 Format(F7.3)
      Write(1,*)'Yearly Average Water Temperature in The Storage'
      Write(1,160)Tmn
  160 Format(F6.2)
      Write(1,*)'Annual Solar Fraction (percent)'
      Write(1,161)F,Ac
  161 Format(F6.2,'  Ac=',F6.2)
      RETURN
      END
C-----------------------------------------------------------------
      SUBROUTINE FAIR(ke,ka,Pi,Pik,Fa,Fake,Faka)
      Complex Fake(20),Faka(20),Fasum
      Real Fa(12),Pik(13)
      DO 81 K=-1,ke,-1
      k1=-1*k
      FASUM=CMPLX(0.,0.)
      DO 80 J=1,12
      Fsr=Fa(j)*(Sin(Pik(J+1)*k)-Sin(Pik(J)*k))
      Fsc=Fa(j)*(Cos(Pik(J+1)*k)-COS(Pik(J)*k))
      FASUM=FASUM+CMPLX(Fsr,Fsc)
   80 CONTINUE
      FAKE(K1)=FASUM/(2*PI*K)
      FAKE(K1)=0.0
   81 CONTINUE
      DO 91 K=1,ka
      FASUM=CMPLX(0.,0.)
      DO 90 J=1,12
      Fsr=Fa(j)*(Sin(Pik(J+1)*k)-Sin(Pik(J)*k))
      Fsc=Fa(j)*(Cos(Pik(J+1)*k)-COS(Pik(J)*k))
      FASUM=FASUM+CMPLX(Fsr,Fsc)
   90 CONTINUE
      FAKA(K)=FASUM/(2*PI*K)
      FAKA(K)=0.0
```

```
  91 CONTINUE
      Return
      End
C*****************************************************
      SUBROUTINE QNET(ke,ka,Pi,Pik,Qn,Qke,Qka)
      Complex Qke(20),Qka(20),Qsum
      Real Qn(12),Pik(13)
      DO 30 K=-1,ke,-1
      QSUM=CMPLX(0.,0.)
      DO 20 J=1,12
      Qsr=Qn(J)*(Sin(Pik(J+1)*k)-Sin(Pik(J)*k))
      Qsc=Qn(J)*(Cos(Pik(J+1)*k)-Cos(Pik(J)*k))
      Qsum=Qsum+Cmplx(Qsr,Qsc)
  20 CONTINUE
      k1=-1*k
      QKE(k1)=QSUM/(2*Pi*k)
  30 CONTINUE
      DO 31 K=1,ka
      QSUM=CMPLX(0.,0.)
      DO 21 J=1,12
      Qsr=Qn(J)*(Sin(Pik(J+1)*k)-Sin(Pik(J)*k))
      Qsc=Qn(J)*(Cos(Pik(J+1)*k)-Cos(Pik(J)*k))
      Qsum=Qsum+Cmplx(Qsr,Qsc)
  21 CONTINUE
      QKA(K)=Qsum/(2*Pi*k)
  31 CONTINUE
      Return
      End
C****************************************************************************************************
C* THIS SUBROUTINE CALCULATES THE MONTHLY AVERAGE                                          *
C* TRANSMITTANCE-ABSORBTANCE PRODUCT AND COLLECTOR                                         *
C* HEAT REMOVAL FACTOR                                                                    *
C****************************************************************************************************
      SUBROUTINE TAGALFA(il,Fb,Tb,Ac,Ref,HH,DLTA,RBA,M,TALF,FR)
      REAL HH(12),DLTA(12),RBA(12),TALF(12),REF(12),TALB(12),MD,MDC
      REAL UL(4),TALN(4),B0(4),Fb(5),Tb(5)
      DATA MD,PK,W/0.01,45.,0.088/
      DATA UL/7.4,4.5,5.,3.2/,TALN/.89,.76,.80,.74/,B0/.078,.15,.11,.16/
      DATA DLT,DS,CB,CPC,HC/0.0005,0.01,300.,3300.,300./
      DATA PI/3.1415/
      FIB=Fb(il)*PI/180.
      BTB=TB(il)*PI/180.
      TR=89.8-0.5788*TB(il)+0.002693*TB(il)*TB(il)
      TETR=PI*TR/180.
      TALR=TALN(M)*(1.+B0(M)*(1.-1./COS(TETR)))
      TETD=PI/3.
      TALD=TALN(M)*(1.+B0(M)*(1.-1./COS(TETD)))
      DO 1 I=1,12
      TB1=ACOS(COS(FIB-BTB)*COS(DLTA(I))*COS(5.*PI/24.)+
     *SIN(FIB-BTB)*SIN(DLTA(I)))
      TALB(I)=TALN(M)*(1.+B0(M)*(1.-1./COS(TB1)))
      TAL1=(1.-HH(I))*RBA(I)*TALB(I)+HH(I)*TALD*(1.+COS(BTB))
     */2.+REF(I)*TALR*(1.-COS(BTB))/2.
      TAL2=(1.-HH(I))*RBA(I)+HH(I)*(1.+COS(BTB))/2.+REF(I)*(1.-COS(BTB)
     *)/2.
      TALF(I)=TAL1/TAL2
   1 CONTINUE
      MDC=MD*AC
      QM=SQRT(UL(M)/(PK*DLT))
      FL=TANH(QM*(W-DS)/2.)/(QM*(W-DS)/2.0)
      FP=1./(UL(M)*W*(1./(UL(M)*(DS+(W-DS)*FL))+1.0/CB+1./(PI*DS*HC)))
```

```
   FR=MDC*CPC/(AC*UL(M))*(1.-EXP(-1.*AC*UL(M)*FP/(MDC*CPC)))
   RETURN
   END
C********************************************************************************************
C THIS SUBROUTINE SUBPROGRAM CALCULATES MONTHLY AVERAGE RADIATION
C ON TILTED SURFACE
C********************************************************************************************
   SUBROUTINE COLRAD(il,Fb,Tb,HH,DLTA,RBA,HT,RT,RN,HO,KTM,
  &AA,BB,CC,RR,Ref,IHM)
   REAL WS(12),HDH(12),WSP1(12),WSP2(12),WSP(12),A1(12),
  $B1(12),RD(12),IHM(5,12),RBB(12)
   REAL HH(12),DLTA(12),RBA(12),HT(12),RT(12),RN(12),HO(12),KTM(12)
  $,AA(12),BB(12),CC(12),RR(12),REF(12),Fb(5),Tb(5)
   INTEGER NAV(12)
   DATA WW,PI/.0,3.1415/
   DATA NAV/198,228,258,288,318,334,17,47,75,105,135,162/
   FIB=Fb(il)*PI/180.
   BTB=TB(il)*PI/180.
   DO 3 I=1,12
   DLTA(I)=23.45*SIN(2.*PI/365.*(284.+NAV(I)))*PI/180.
   WS(I)=ACOS(-1.*TAN(FIB)*TAN(DLTA(I)))
   HQ=24*3600.*1353./PI*(1.+0.033*COS(2.*PI*NAV(I)/365.))*(COS(FIB)*
  $COS(DLTA(I))*SIN(WS(I))+WS(I)*SIN(FIB)*SIN(DLTA(I)))
   HO(I)=ABS(HQ)/10**6
   KTM(I)=IHM(il,I)/HO(I)
   HDH(I)=0.775+0.00653*(WS(I)*180./PI-90.)-(0.505+0.00455*
  $(WS(I)*180./PI-90.))*COS((115*KTM(I)-103.)*PI/180.)
   WSP1(I)=ACOS(-1.*TAN(FIB)*TAN(DLTA(I)))
   WSP2(I)=ACOS(-1.*TAN(FIB-BTB)*TAN(DLTA(I)))
   IF(WSP1(I).GE.WSP2(I)) GO TO 1
   WSP(I)=WSP1(I)
   GO TO 2
  1 WSP(I)=WSP2(I)
  2 RA=(COS(FIB-BTB)*COS(DLTA(I))*SIN(WSP(I))+WSP(I)*
  $SIN(FIB-BTB)*SIN(DLTA(I)))/(COS(FIB)*COS(DLTA(I))*SIN(WS(I))+
  $WS(I)*SIN(FIB)*SIN(DLTA(I)))
   RBA(I)=ABS(RA)
   RR(I)=(1.-HDH(I))*RBA(I)+HDH(I)*(1.+COS(BTB))/2.+REF(I)*
  &(1.-COS(BTB))/2.
   HT(I)=RR(I)*IHM(il,I)
   A1(I)=0.409+0.5016*SIN(WS(I)-(PI/3.))
   B1(I)=0.6609-0.4767*SIN(WS(I)-(PI/3.))
   RD(I)=PI/24.*(COS(WW)-COS(WS(I)))/(SIN(WS(I))-WS(I)*COS(WS(I)))
   RT(I)=PI/24.*(A1(I)+B1(I)*COS(WW))*(COS(WW)-COS(WS(I)))/
  *(SIN(WS(I))-WS(I)*COS(WS(I)))
   HH(I)=1.188-2.272*KTM(I)+9.473*KTM(I)**2-21.865*KTM(I)**3+
  *14.648*KTM(I)**4
   IF(KTM(I).GE.0.8) HH(I)=0.2
   IF(KTM(I).LE.0.17) HH(I)=0.99
   RBB(I)=ABS((COS(FIB-BTB)*COS(DLTA(I))*COS(WW)+
  *SIN(FIB-BTB)*SIN(DLTA(I)))/(COS(FIB)*COS(DLTA(I))*COS(WW)+
  *SIN(FIB)*SIN(DLTA(I))))
   RN(I)=(1.-RD(I)*HH(I)/RT(I))*RBB(I)+RD(I)/RT(I)*HH(I)*(1.+
  *COS(BTB))/2.+REF(I)*(1.-COS(BTB))/2.
   AA(I)=2.943-9.271*KTM(I)+4.031*KTM(I)**2.
   BB(I)=-4.345+8.853*KTM(I)-3.602*KTM(I)**2.
   CC(I)=-0.170-0.306*KTM(I)+2.936*KTM(I)**2.
  3 CONTINUE
   RETURN
   END
```

```
C************************************************************************
c  This subroutine writes all reqired outputs
C************************************************************************
     SUBROUTINE RESULT(il,M,K,Tb,Ac,R,NOH,Ta,Sl,Aylr,Typ,City,IHM
    $,KTM,RR,RN,XCB,FBAR,Ef,Tnew,Ts1,Tx,QSY,WY,QLT,QLY,QY,R1,R2,R4,R5)
C************************************************************************
     REAL IHM(5,12),KTM(12),RR(12),RN(12),XCB(12),FBAR(12)
     Real Ef(12),TNEW(12),Tx(12),Ts1(12),NOH,Ta(5,12),Tb(5)
     CHARACTER*33 AYLR(12),SL(5),TYP(10),City(5)
     WRITE(1,1)TYP(M),City(il),Noh,Sl(K),Tb(il),Ac,R
     WRITE(6,1)TYP(M),City(il),Noh,Sl(K),Tb(il),Ac,R
   1 FORMAT(' Collector=',A33,/' City  = ',A9,'   House Nmr.=',f5.0,/
    &' Earth = ',A6,'      Beta      = ',F6.2,/
    &' Ac    =',F4.0,' m2','      Store Rad.=',F7.2,' m')
     WRITE(1,2)
     WRITE(6,2)
   2 FORMAT(1X,69('_'),/1X,'| ','MNTH',' TA ',' H(mj/m2)',' KT  ','  R'
    $,3X,'  RN  ','  XC ','  Fib','  Efncy ',' Tstr','    Tx ',2X,'|',
    $/1X,'|',67('_'),'|')
     DO 4 I=1,12
     WRITE(1,3)AYLR(I),Ta(il,I),IHM(il,I),KTM(I),RR(I),RN(I),XCB(I)
    $,FBAR(I),Ef(I),Tnew(I),Tx(I)
     WRITE(6,3)AYLR(I),Ta(il,I),IHM(il,I),KTM(I),RR(I),RN(I),XCB(I)
    $,FBAR(I),Ef(I),Tnew(I),Tx(I)
   3 FORMAT(1X,'|',1X,A3,8(1X,F5.2),2X,F5.2,2X,F5.2,1X,'|')
   4 CONTINUE
     WRITE(1,5)
     WRITE(6,5)
   5 FORMAT(1X,'|',67('_'),'|',/' |',1X,' QSY',4X,'WY',
    $4X,'QLT',4X,'QLY',4X,'QY',4X,'R1',4X,'R2',4X,'R4',4X,'R5',5X,
    $'COP',4X,'|'/1X,'|',67('_'),'|')
     IF(WY.EQ.0.0) GO TO 7
     COP=QLY/WY
     WRITE(1,6)QSY,WY,QLT,QLY,QY,R1,R2,R4,R5,COP
     WRITE(6,6)QSY,WY,QLT,QLY,QY,R1,R2,R4,R5,COP
   6 FORMAT(1X,'| ',F5.3,1X,F5.4,2X,F5.3,2X,F5.3,2X,F5.3,4(1X,
    $F5.3),2X,F6.2,2X,'|',/1X,'|',67('-'),'|')
     GO TO 9
   7 WRITE(1,8)QSY,WY,QLT,QLY,QY,R1,R2,R4,R5
     WRITE(6,8)QSY,WY,QLT,QLY,QY,R1,R2,R4,R5
   8 FORMAT(1X,'| ',F5.3,1X,F5.4,2X,F5.3,2X,F5.3,2X,F5.3,
    &4(1X,F5.3),1X,' NO PUMP',' |',/,1X,'|',67('-'),'|')
   9 RETURN
     END


* ======================================================================
* NIST Guide to Available Math Software.
* Fullsource for module CBESH from package AMOS.
* Retrieved from NETLIB on Mon Nov 24 01:51:01 1997.
* ======================================================================
      SUBROUTINE CBESH(Z, FNU, KODE, M, N, CY, NZ, IERR)
C***BEGIN PROLOGUE  CBESH
C***DATE WRITTEN   830501   (YYMMDD)
C***REVISION DATE  890801   (YYMMDD)
C***CATEGORY NO.  B5K
C***KEYWORDS  H-BESSEL FUNCTIONS,BESSEL FUNCTIONS OF COMPLEX
ARGUMENT,
C             BESSEL FUNCTIONS OF THIRD KIND,HANKEL FUNCTIONS
C***AUTHOR  AMOS, DONALD E., SANDIA NATIONAL LABORATORIES
```

```
C***PURPOSE  TO COMPUTE THE H-BESSEL FUNCTIONS OF A COMPLEX
ARGUMENT
C***DESCRIPTION
C
C       ON KODE=1, CBESH COMPUTES AN N MEMBER SEQUENCE OF COMPLEX
C       HANKEL (BESSEL) FUNCTIONS CY(J)=H(M,FNU+J-1,Z) FOR KINDS M=1
C       OR 2, REAL, NONNEGATIVE ORDERS FNU+J-1, J=1,...,N, AND COMPLEX
C       Z.NE.CMPLX(0.0E0,0.0E0) IN THE CUT PLANE -PI.LT.ARG(Z).LE.PI.
C       ON KODE=2, CBESH COMPUTES THE SCALED HANKEL FUNCTIONS
C
C       CY(I)=H(M,FNU+J-1,Z)*EXP(-MM*Z*I)       MM=3-2M,      I**2=-1.
C
C       WHICH REMOVES THE EXPONENTIAL BEHAVIOR IN BOTH THE UPPER
C       AND LOWER HALF PLANES. DEFINITIONS AND NOTATION ARE FOUND IN
C       THE NBS HANDBOOK OF MATHEMATICAL FUNCTIONS (REF. 1).
C
C       INPUT
C         Z     - Z=CMPLX(X,Y), Z.NE.CMPLX(0.,0.),-PI.LT.ARG(Z).LE.PI
C         FNU   - ORDER OF INITIAL H FUNCTION, FNU.GE.0.0E0
C         KODE  - A PARAMETER TO INDICATE THE SCALING OPTION
C                KODE= 1  RETURNS
C                      CY(J)=H(M,FNU+J-1,Z),      J=1,...,N
C                    = 2  RETURNS
C                      CY(J)=H(M,FNU+J-1,Z)*EXP(-I*Z*(3-2M))
C                         J=1,...,N  ,  I**2=-1
C         M     - KIND OF HANKEL FUNCTION, M=1 OR 2
C         N     - NUMBER OF MEMBERS OF THE SEQUENCE, N.GE.1
C
C       OUTPUT
C         CY    - A COMPLEX VECTOR WHOSE FIRST N COMPONENTS CONTAIN
C                VALUES FOR THE SEQUENCE
C                CY(J)=H(M,FNU+J-1,Z)  OR
C                CY(J)=H(M,FNU+J-1,Z)*EXP(-I*Z*(3-2M))  J=1,...,N
C                DEPENDING ON KODE, I**2=-1.
C         NZ    - NUMBER OF COMPONENTS SET TO ZERO DUE TO UNDERFLOW,
C                NZ= 0   , NORMAL RETURN
C                NZ.GT.0 , FIRST NZ COMPONENTS OF CY SET TO ZERO
C                          DUE TO UNDERFLOW, CY(J)=CMPLX(0.0,0.0)
C                          J=1,...,NZ WHEN Y.GT.0.0 AND M=1 OR
C                          Y.LT.0.0 AND M=2. FOR THE COMPLMENTARY
C                          HALF PLANES, NZ STATES ONLY THE NUMBER
C                          OF UNDERFLOWS.
C         IERR  -ERROR FLAG
C                IERR=0, NORMAL RETURN - COMPUTATION COMPLETED
C                IERR=1, INPUT ERROR   - NO COMPUTATION
C                IERR=2, OVERFLOW      - NO COMPUTATION, FNU+N-1 TOO
C                       LARGE OR CABS(Z) TOO SMALL OR BOTH
C                IERR=3, CABS(Z) OR FNU+N-1 LARGE - COMPUTATION DONE
C                       BUT LOSSES OF SIGNIFCANCE BY ARGUMENT
C                       REDUCTION PRODUCE LESS THAN HALF OF MACHINE
C                       ACCURACY
C                IERR=4, CABS(Z) OR FNU+N-1 TOO LARGE - NO COMPUTA-
C                       TION BECAUSE OF COMPLETE LOSSES OF SIGNIFI-
C                       CANCE BY ARGUMENT REDUCTION
C                IERR=5, ERROR              - NO COMPUTATION,
C                       ALGORITHM TERMINATION CONDITION NOT MET
C***LONG DESCRIPTION
C       THE COMPUTATION IS CARRIED OUT BY THE RELATION
C
C       H(M,FNU,Z)=(1/MP)*EXP(-MP*FNU)*K(FNU,Z*EXP(-MP))
C          MP=MM*HPI*I,  MM=3-2*M,  HPI=PI/2,  I**2=-1
```

```
C
C       FOR M=1 OR 2 WHERE THE K BESSEL FUNCTION IS COMPUTED FOR THE
C       RIGHT HALF PLANE RE(Z).GE.0.0. THE K FUNCTION IS CONTINUED
C       TO THE LEFT HALF PLANE BY THE RELATION
C
C       K(FNU,Z*EXP(MP)) = EXP(-MP*FNU)*K(FNU,Z)-MP*I(FNU,Z)
C       MP=MR*PI*I, MR=+1 OR -1, RE(Z).GT.0, I**2=-1
C
C       WHERE I(FNU,Z) IS THE I BESSEL FUNCTION.
C
C       EXPONENTIAL DECAY OF H(M,FNU,Z) OCCURS IN THE UPPER HALF Z
C       PLANE FOR M=1 AND THE LOWER HALF Z PLANE FOR M=2.  EXPONENTIAL
C       GROWTH OCCURS IN THE COMPLEMENTARY HALF PLANES.  SCALING
C       BY EXP(-MM*Z*I) REMOVES THE EXPONENTIAL BEHAVIOR IN THE
C       WHOLE Z PLANE FOR Z TO INFINITY.
C
C       FOR NEGATIVE ORDERS,THE FORMULAE
C
C           H(1,-FNU,Z) = H(1,FNU,Z)*CEXP( PI*FNU*I)
C           H(2,-FNU,Z) = H(2,FNU,Z)*CEXP(-PI*FNU*I)
C                 I**2=-1
C
C       CAN BE USED.
C
C       IN MOST COMPLEX VARIABLE COMPUTATION, ONE MUST EVALUATE ELE-
C       MENTARY FUNCTIONS. WHEN THE MAGNITUDE OF Z OR FNU+N-1 IS
C       LARGE, LOSSES OF SIGNIFICANCE BY ARGUMENT REDUCTION OCCUR.
C       CONSEQUENTLY, IF EITHER ONE EXCEEDS U1=SQRT(0.5/UR), THEN
C       LOSSES EXCEEDING HALF PRECISION ARE LIKELY AND AN ERROR FLAG
C       IERR=3 IS TRIGGERED WHERE UR=R1MACH(4)=UNIT ROUNDOFF. ALSO
C       IF EITHER IS LARGER THAN U2=0.5/UR, THEN ALL SIGNIFICANCE IS
C       LOST AND IERR=4. IN ORDER TO USE THE INT FUNCTION, ARGUMENTS
C       MUST BE FURTHER RESTRICTED NOT TO EXCEED THE LARGEST MACHINE
C       INTEGER, U3=I1MACH(9). THUS, THE MAGNITUDE OF Z AND FNU+N-1 IS
C       RESTRICTED BY MIN(U2,U3). ON 32 BIT MACHINES, U1,U2, AND U3
C       ARE APPROXIMATELY 2.0E+3, 4.2E+6, 2.1E+9 IN SINGLE PRECISION
C       ARITHMETIC AND 1.3E+8, 1.8E+16, 2.1E+9 IN DOUBLE PRECISION
C       ARITHMETIC RESPECTIVELY. THIS MAKES U2 AND U3 LIMITING IN
C       THEIR RESPECTIVE ARITHMETICS. THIS MEANS THAT ONE CAN EXPECT
C       TO RETAIN, IN THE WORST CASES ON 32 BIT MACHINES, NO DIGITS
C       IN SINGLE AND ONLY 7 DIGITS IN DOUBLE PRECISION ARITHMETIC.
C       SIMILAR CONSIDERATIONS HOLD FOR OTHER MACHINES.
C
C       THE APPROXIMATE RELATIVE ERROR IN THE MAGNITUDE OF A COMPLEX
C       BESSEL FUNCTION CAN BE EXPRESSED BY P*10**S WHERE P=MAX(UNIT
C       ROUNDOFF,1.0E-18) IS THE NOMINAL PRECISION AND 10**S REPRE-
C       SENTS THE INCREASE IN ERROR DUE TO ARGUMENT REDUCTION IN THE
C       ELEMENTARY FUNCTIONS. HERE, S=MAX(1,ABS(LOG10(CABS(Z))),
C       ABS(LOG10(FNU))) APPROXIMATELY (I.E. S=MAX(1,ABS(EXPONENT OF
C       CABS(Z),ABS(EXPONENT OF FNU)) ). HOWEVER, THE PHASE ANGLE MAY
C       HAVE ONLY ABSOLUTE ACCURACY. THIS IS MOST LIKELY TO OCCUR WHEN
C       ONE COMPONENT (IN ABSOLUTE VALUE) IS LARGER THAN THE OTHER BY
C       SEVERAL ORDERS OF MAGNITUDE. IF ONE COMPONENT IS 10**K LARGER
C       THAN THE OTHER, THEN ONE CAN EXPECT ONLY MAX(ABS(LOG10(P))-K,
C       0) SIGNIFICANT DIGITS; OR, STATED ANOTHER WAY, WHEN K EXCEEDS
C       THE EXPONENT OF P, NO SIGNIFICANT DIGITS REMAIN IN THE SMALLER
C       COMPONENT. HOWEVER, THE PHASE ANGLE RETAINS ABSOLUTE
ACCURACY
C       BECAUSE, IN COMPLEX ARITHMETIC WITH PRECISION P, THE SMALLER
C       COMPONENT WILL NOT (AS A RULE) DECREASE BELOW P TIMES THE
C       MAGNITUDE OF THE LARGER COMPONENT. IN THESE EXTREME CASES,
```

```
C         THE PRINCIPAL PHASE ANGLE IS ON THE ORDER OF +P, -P, PI/2-P,
C         OR -PI/2+P.
C***REFERENCES  HANDBOOK OF MATHEMATICAL FUNCTIONS BY M. ABRAMOWITZ
C                 AND I. A. STEGUN, NBS AMS SERIES 55, U.S. DEPT. OF
C                 COMMERCE, 1955.
C
C               COMPUTATION OF BESSEL FUNCTIONS OF COMPLEX ARGUMENT
C                 BY D. E. AMOS, SAND83-0083, MAY, 1983.
C
C               COMPUTATION OF BESSEL FUNCTIONS OF COMPLEX ARGUMENT
C                 AND LARGE ORDER BY D. E. AMOS, SAND83-0643, MAY, 1983
C
C               A SUBROUTINE PACKAGE FOR BESSEL FUNCTIONS OF A COMPLEX
C                 ARGUMENT AND NONNEGATIVE ORDER BY D. E. AMOS, SAND85-
C                 1018, MAY, 1985
C
C               A PORTABLE PACKAGE FOR BESSEL FUNCTIONS OF A COMPLEX
C                 ARGUMENT AND NONNEGATIVE ORDER BY D. E. AMOS, TRANS.
C                 MATH. SOFTWARE, 1986
C***ROUTINES CALLED  CACON,CBKNU,CBUNK,CUOIK,I1MACH,R1MACH
C***END PROLOGUE  CBESH
      COMPLEX CY, Z, ZN, ZT, CSGN
      REAL AA, ALIM, ALN, ARG, AZ, CPN, DIG, ELIM, FMM, FN, FNU, FNUL,
     * HPI, RHPI, RL, R1M5, SGN, SPN, TOL, UFL, XN, XX, YN, YY, R1MACH,
     * BB, ASCLE, RTOL, ATOL
      INTEGER I, IERR, INU, INUH, IR, K, KODE, K1, K2, M,
     * MM, MR, N, NN, NUF, NW, NZ, I1MACH
      DIMENSION CY(N)
      DATA HPI /1.57079632679489662E0/
C***FIRST EXECUTABLE STATEMENT  CBESH
      NZ=0
      XX = REAL(Z)
      YY = AIMAG(Z)
      IERR = 0
      IF (XX.EQ.0.0E0 .AND. YY.EQ.0.0E0) IERR=1
      IF (FNU.LT.0.0E0) IERR=1
      IF (M.LT.1 .OR. M.GT.2) IERR=1
      IF (KODE.LT.1 .OR. KODE.GT.2) IERR=1
      IF (N.LT.1) IERR=1
      IF (IERR.NE.0) RETURN
      NN = N
C-----------------------------------------------------------------------
C     SET PARAMETERS RELATED TO MACHINE CONSTANTS.
C     TOL IS THE APPROXIMATE UNIT ROUNDOFF LIMITED TO 1.0E-18.
C     ELIM IS THE APPROXIMATE EXPONENTIAL OVER- AND UNDERFLOW LIMIT.
C     EXP(-ELIM).LT.EXP(-ALIM)=EXP(-ELIM)/TOL    AND
C     EXP(ELIM).GT.EXP(ALIM)=EXP(ELIM)*TOL       ARE INTERVALS NEAR
C     UNDERFLOW AND OVERFLOW LIMITS WHERE SCALED ARITHMETIC IS DONE.
C     RL IS THE LOWER BOUNDARY OF THE ASYMPTOTIC EXPANSION FOR LARGE Z.
C     DIG = NUMBER OF BASE 10 DIGITS IN TOL = 10**(-DIG).
C     FNUL IS THE LOWER BOUNDARY OF THE ASYMPTOTIC SERIES FOR LARGE FNU
C-----------------------------------------------------------------------
      TOL = AMAX1(R1MACH(4),1.0E-18)
      K1 = I1MACH(12)
      K2 = I1MACH(13)
      R1M5 = R1MACH(5)
      K = MIN0(IABS(K1),IABS(K2))
      ELIM = 2.303E0*(FLOAT(K)*R1M5-3.0E0)
      K1 = I1MACH(11) - 1
      AA = R1M5*FLOAT(K1)
      DIG = AMIN1(AA,18.0E0)
```

```
      AA = AA*2.303E0
      ALIM = ELIM + AMAX1(-AA,-41.45E0)
      FNUL = 10.0E0 + 6.0E0*(DIG-3.0E0)
      RL = 1.2E0*DIG + 3.0E0
      FN = FNU + FLOAT(NN-1)
      MM = 3 - M - M
      FMM = FLOAT(MM)
      ZN = Z*CMPLX(0.0E0,-FMM)
      XN = REAL(ZN)
      YN = AIMAG(ZN)
      AZ = CABS(Z)
C-----------------------------------------------------------------------
C     TEST FOR RANGE
C-----------------------------------------------------------------------
      AA = 0.5E0/TOL
      BB=FLOAT(I1MACH(9))*0.5E0
      AA=AMIN1(AA,BB)
      IF(AZ.GT.AA) GO TO 240
      IF(FN.GT.AA) GO TO 240
      AA=SQRT(AA)
      IF(AZ.GT.AA) IERR=3
      IF(FN.GT.AA) IERR=3
C-----------------------------------------------------------------------
C     OVERFLOW TEST ON THE LAST MEMBER OF THE SEQUENCE
C-----------------------------------------------------------------------
      UFL = R1MACH(1)*1.0E+3
      IF (AZ.LT.UFL) GO TO 220
      IF (FNU.GT.FNUL) GO TO 90
      IF (FN.LE.1.0E0) GO TO 70
      IF (FN.GT.2.0E0) GO TO 60
      IF (AZ.GT.TOL) GO TO 70
      ARG = 0.5E0*AZ
      ALN = -FN*ALOG(ARG)
      IF (ALN.GT.ELIM) GO TO 220
      GO TO 70
   60 CONTINUE
      CALL CUOIK(ZN, FNU, KODE, 2, NN, CY, NUF, TOL, ELIM, ALIM)
      IF (NUF.LT.0) GO TO 220
      NZ = NZ + NUF
      NN = NN - NUF
C-----------------------------------------------------------------------
C     HERE NN=N OR NN=0 SINCE NUF=0,NN, OR -1 ON RETURN FROM CUOIK
C     IF NUF=NN, THEN CY(I)=CZERO FOR ALL I
C-----------------------------------------------------------------------
      IF (NN.EQ.0) GO TO 130
   70 CONTINUE
      IF ((XN.LT.0.0E0) .OR. (XN.EQ.0.0E0 .AND. YN.LT.0.0E0 .AND.
     * M.EQ.2)) GO TO 80
C-----------------------------------------------------------------------
C     RIGHT HALF PLANE COMPUTATION, XN.GE.0. .AND. (XN.NE.0. .OR.
C     YN.GE.0. .OR. M=1)
C-----------------------------------------------------------------------
      CALL CBKNU(ZN, FNU, KODE, NN, CY, NZ, TOL, ELIM, ALIM)
      GO TO 110
C-----------------------------------------------------------------------
C     LEFT HALF PLANE COMPUTATION
C-----------------------------------------------------------------------
   80 CONTINUE
      MR = -MM
      CALL CACON(ZN, FNU, KODE, MR, NN, CY, NW, RL, FNUL, TOL, ELIM,
     * ALIM)
```

```
      IF (NW.LT.0) GO TO 230
      NZ=NW
      GO TO 110
   90 CONTINUE
C-----------------------------------------------------------------------
C     UNIFORM ASYMPTOTIC EXPANSIONS FOR FNU.GT.FNUL
C-----------------------------------------------------------------------
      MR = 0
      IF ((XN.GE.0.0E0) .AND. (XN.NE.0.0E0 .OR. YN.GE.0.0E0 .OR.
     * M.NE.2)) GO TO 100
      MR = -MM
      IF (XN.EQ.0.0E0 .AND. YN.LT.0.0E0) ZN = -ZN
  100 CONTINUE
      CALL CBUNK(ZN, FNU, KODE, MR, NN, CY, NW, TOL, ELIM, ALIM)
      IF (NW.LT.0) GO TO 230
      NZ = NZ + NW
  110 CONTINUE
C-----------------------------------------------------------------------
C     H(M,FNU,Z) = -FMM*(I/HPI)*(ZT**FNU)*K(FNU,-Z*ZT)
C
C     ZT=EXP(-FMM*HPI*I) = CMPLX(0.0,-FMM), FMM=3-2*M, M=1,2
C-----------------------------------------------------------------------
      SGN = SIGN(HPI,-FMM)
C-----------------------------------------------------------------------
C     CALCULATE EXP(FNU*HPI*I) TO MINIMIZE LOSSES OF SIGNIFICANCE
C     WHEN FNU IS LARGE
C-----------------------------------------------------------------------
      INU = INT(FNU)
      INUH = INU/2
      IR = INU - 2*INUH
      ARG = (FNU-FLOAT(INU-IR))*SGN
      RHPI = 1.0E0/SGN
      CPN = RHPI*COS(ARG)
      SPN = RHPI*SIN(ARG)
C     ZN = CMPLX(-SPN,CPN)
      CSGN = CMPLX(-SPN,CPN)
C     IF (MOD(INUH,2).EQ.1) ZN = -ZN
      IF (MOD(INUH,2).EQ.1) CSGN = -CSGN
      ZT = CMPLX(0.0E0,-FMM)
      RTOL = 1.0E0/TOL
      ASCLE = UFL*RTOL
      DO 120 I=1,NN
C       CY(I) = CY(I)*ZN
C       ZN = ZN*ZT
        ZN=CY(I)
        AA=REAL(ZN)
        BB=AIMAG(ZN)
        ATOL=1.0E0
        IF (AMAX1(ABS(AA),ABS(BB)).GT.ASCLE) GO TO 125
          ZN = ZN*CMPLX(RTOL,0.0E0)
          ATOL = TOL
  125   CONTINUE
        ZN = ZN*CSGN
        CY(I) = ZN*CMPLX(ATOL,0.0E0)
        CSGN = CSGN*ZT
  120 CONTINUE
      RETURN
  130 CONTINUE
      IF (XN.LT.0.0E0) GO TO 220
      RETURN
  220 CONTINUE
```

```
      IERR=2
      NZ=0
      RETURN
  230 CONTINUE
      IF(NW.EQ.(-1)) GO TO 220
      NZ=0
      IERR=5
      RETURN
  240 CONTINUE
      NZ=0
      IERR=4
      RETURN
      END
      SUBROUTINE CACAI(Z, FNU, KODE, MR, N, Y, NZ, RL, TOL, ELIM, ALIM)
C***BEGIN PROLOGUE  CACAI
C***REFER TO  CAIRY
C
C     CACAI APPLIES THE ANALYTIC CONTINUATION FORMULA
C
C         K(FNU,ZN*EXP(MP))=K(FNU,ZN)*EXP(-MP*FNU) - MP*I(FNU,ZN)
C                 MP=PI*MR*CMPLX(0.0,1.0)
C
C     TO CONTINUE THE K FUNCTION FROM THE RIGHT HALF TO THE LEFT
C     HALF Z PLANE FOR USE WITH CAIRY WHERE FNU=1/3 OR 2/3 AND N=1.
C     CACAI IS THE SAME AS CACON WITH THE PARTS FOR LARGER ORDERS AND
C     RECURRENCE REMOVED. A RECURSIVE CALL TO CACON CAN RESULT IF
CACON
C     IS CALLED FROM CAIRY.
C
C***ROUTINES CALLED  CASYI,CBKNU,CMLRI,CSERI,CS1S2,R1MACH
C***END PROLOGUE  CACAI
      COMPLEX CSGN, CSPN, C1, C2, Y, Z, ZN, CY
      REAL ALIM, ARG, ASCLE, AZ, CPN, DFNU, ELIM, FMR, FNU, PI, RL,
     * SGN, SPN, TOL, YY, R1MACH
      INTEGER INU, IUF, KODE, MR, N, NN, NW, NZ
      DIMENSION Y(N), CY(2)
      DATA PI / 3.14159265358979324E0 /
      NZ = 0
      ZN = -Z
      AZ = CABS(Z)
      NN = N
      DFNU = FNU + FLOAT(N-1)
      IF (AZ.LE.2.0E0) GO TO 10
      IF (AZ*AZ*0.25E0.GT.DFNU+1.0E0) GO TO 20
   10 CONTINUE
C-----------------------------------------------------------------------
C     POWER SERIES FOR THE I FUNCTION
C-----------------------------------------------------------------------
      CALL CSERI(ZN, FNU, KODE, NN, Y, NW, TOL, ELIM, ALIM)
      GO TO 40
   20 CONTINUE
      IF (AZ.LT.RL) GO TO 30
C-----------------------------------------------------------------------
C     ASYMPTOTIC EXPANSION FOR LARGE Z FOR THE I FUNCTION
C-----------------------------------------------------------------------
      CALL CASYI(ZN, FNU, KODE, NN, Y, NW, RL, TOL, ELIM, ALIM)
      IF (NW.LT.0) GO TO 70
      GO TO 40
   30 CONTINUE
C-----------------------------------------------------------------------
C     MILLER ALGORITHM NORMALIZED BY THE SERIES FOR THE I FUNCTION
```

```
C-----------------------------------------------------------------------
      CALL CMLRI(ZN, FNU, KODE, NN, Y, NW, TOL)
      IF(NW.LT.0) GO TO 70
   40 CONTINUE
C-----------------------------------------------------------------------
C     ANALYTIC CONTINUATION TO THE LEFT HALF PLANE FOR THE K FUNCTION
C-----------------------------------------------------------------------
      CALL CBKNU(ZN, FNU, KODE, 1, CY, NW, TOL, ELIM, ALIM)
      IF (NW.NE.0) GO TO 70
      FMR = FLOAT(MR)
      SGN = -SIGN(PI,FMR)
      CSGN = CMPLX(0.0E0,SGN)
      IF (KODE.EQ.1) GO TO 50
      YY = -AIMAG(ZN)
      CPN = COS(YY)
      SPN = SIN(YY)
      CSGN = CSGN*CMPLX(CPN,SPN)
   50 CONTINUE
C-----------------------------------------------------------------------
C     CALCULATE CSPN=EXP(FNU*PI*I) TO MINIMIZE LOSSES OF SIGNIFICANCE
C     WHEN FNU IS LARGE
C-----------------------------------------------------------------------
      INU = INT(FNU)
      ARG = (FNU-FLOAT(INU))*SGN
      CPN = COS(ARG)
      SPN = SIN(ARG)
      CSPN = CMPLX(CPN,SPN)
      IF (MOD(INU,2).EQ.1) CSPN = -CSPN
      C1 = CY(1)
      C2 = Y(1)
      IF (KODE.EQ.1) GO TO 60
      IUF = 0
      ASCLE = 1.0E+3*R1MACH(1)/TOL
      CALL CS1S2(ZN, C1, C2, NW, ASCLE, ALIM, IUF)
      NZ = NZ + NW
   60 CONTINUE
      Y(1) = CSPN*C1 + CSGN*C2
      RETURN
   70 CONTINUE
      NZ = -1
      IF(NW.EQ.(-2)) NZ=-2
      RETURN
      END
      SUBROUTINE CACON(Z, FNU, KODE, MR, N, Y, NZ, RL, FNUL, TOL, ELIM,
     * ALIM)
C***BEGIN PROLOGUE  CACON
C***REFER TO  CBESK,CBESH
C
C     CACON APPLIES THE ANALYTIC CONTINUATION FORMULA
C
C         K(FNU,ZN*EXP(MP))=K(FNU,ZN)*EXP(-MP*FNU) - MP*I(FNU,ZN)
C                 MP=PI*MR*CMPLX(0.0,1.0)
C
C     TO CONTINUE THE K FUNCTION FROM THE RIGHT HALF TO THE LEFT
C     HALF Z PLANE
C
C***ROUTINES CALLED  CBINU,CBKNU,CS1S2,R1MACH
C***END PROLOGUE  CACON
      COMPLEX CK, CONE, CS, CSCL, CSCR, CSGN, CSPN, CSS, CSR, C1, C2,
     * RZ, SC1, SC2, ST, S1, S2, Y, Z, ZN, CY
      REAL ALIM, ARG, ASCLE, AS2, BSCLE, BRY, CPN, C1I, C1M, C1R, ELIM,
```

```
     * FMR, FNU, FNUL, PI, RL, SGN, SPN, TOL, YY, R1MACH
      INTEGER I, INU, IUF, KFLAG, KODE, MR, N, NN, NW, NZ
      DIMENSION Y(N), CY(2), CSS(3), CSR(3), BRY(3)
      DATA PI / 3.14159265358979324E0 /
      DATA CONE / (1.0E0,0.0E0) /
      NZ = 0
      ZN = -Z
      NN = N
      CALL CBINU(ZN, FNU, KODE, NN, Y, NW, RL, FNUL, TOL, ELIM, ALIM)
      IF (NW.LT.0) GO TO 80
C-----------------------------------------------------------------------
C     ANALYTIC CONTINUATION TO THE LEFT HALF PLANE FOR THE K FUNCTION
C-----------------------------------------------------------------------
      NN = MIN0(2,N)
      CALL CBKNU(ZN, FNU, KODE, NN, CY, NW, TOL, ELIM, ALIM)
      IF (NW.NE.0) GO TO 80
      S1 = CY(1)
      FMR = FLOAT(MR)
      SGN = -SIGN(PI,FMR)
      CSGN = CMPLX(0.0E0,SGN)
      IF (KODE.EQ.1) GO TO 10
      YY = -AIMAG(ZN)
      CPN = COS(YY)
      SPN = SIN(YY)
      CSGN = CSGN*CMPLX(CPN,SPN)
   10 CONTINUE
C-----------------------------------------------------------------------
C     CALCULATE CSPN=EXP(FNU*PI*I) TO MINIMIZE LOSSES OF SIGNIFICANCE
C     WHEN FNU IS LARGE
C-----------------------------------------------------------------------
      INU = INT(FNU)
      ARG = (FNU-FLOAT(INU))*SGN
      CPN = COS(ARG)
      SPN = SIN(ARG)
      CSPN = CMPLX(CPN,SPN)
      IF (MOD(INU,2).EQ.1) CSPN = -CSPN
      IUF = 0
      C1 = S1
      C2 = Y(1)
      ASCLE = 1.0E+3*R1MACH(1)/TOL
      IF (KODE.EQ.1) GO TO 20
      CALL CS1S2(ZN, C1, C2, NW, ASCLE, ALIM, IUF)
      NZ = NZ + NW
      SC1 = C1
   20 CONTINUE
      Y(1) = CSPN*C1 + CSGN*C2
      IF (N.EQ.1) RETURN
      CSPN = -CSPN
      S2 = CY(2)
      C1 = S2
      C2 = Y(2)
      IF (KODE.EQ.1) GO TO 30
      CALL CS1S2(ZN, C1, C2, NW, ASCLE, ALIM, IUF)
      NZ = NZ + NW
      SC2 = C1
   30 CONTINUE
      Y(2) = CSPN*C1 + CSGN*C2
      IF (N.EQ.2) RETURN
      CSPN = -CSPN
      RZ = CMPLX(2.0E0,0.0E0)/ZN
      CK = CMPLX(FNU+1.0E0,0.0E0)*RZ
```

```
C-----------------------------------------------------------------------
C    SCALE NEAR EXPONENT EXTREMES DURING RECURRENCE ON K FUNCTIONS
C-----------------------------------------------------------------------
    CSCL = CMPLX(1.0E0/TOL,0.0E0)
    CSCR = CMPLX(TOL,0.0E0)
    CSS(1) = CSCL
    CSS(2) = CONE
    CSS(3) = CSCR
    CSR(1) = CSCR
    CSR(2) = CONE
    CSR(3) = CSCL
    BRY(1) = ASCLE
    BRY(2) = 1.0E0/ASCLE
    BRY(3) = R1MACH(2)
    AS2 = CABS(S2)
    KFLAG = 2
    IF (AS2.GT.BRY(1)) GO TO 40
    KFLAG = 1
    GO TO 50
  40 CONTINUE
    IF (AS2.LT.BRY(2)) GO TO 50
    KFLAG = 3
  50 CONTINUE
    BSCLE = BRY(KFLAG)
    S1 = S1*CSS(KFLAG)
    S2 = S2*CSS(KFLAG)
    CS = CSR(KFLAG)
    DO 70 I=3,N
      ST = S2
      S2 = CK*S2 + S1
      S1 = ST
      C1 = S2*CS
      ST = C1
      C2 = Y(I)
      IF (KODE.EQ.1) GO TO 60
      IF (IUF.LT.0) GO TO 60
      CALL CS1S2(ZN, C1, C2, NW, ASCLE, ALIM, IUF)
      NZ = NZ + NW
      SC1 = SC2
      SC2 = C1
      IF (IUF.NE.3) GO TO 60
      IUF = -4
      S1 = SC1*CSS(KFLAG)
      S2 = SC2*CSS(KFLAG)
      ST = SC2
  60   CONTINUE
      Y(I) = CSPN*C1 + CSGN*C2
      CK = CK + RZ
      CSPN = -CSPN
      IF (KFLAG.GE.3) GO TO 70
      C1R = REAL(C1)
      C1I = AIMAG(C1)
      C1R = ABS(C1R)
      C1I = ABS(C1I)
      C1M = AMAX1(C1R,C1I)
      IF (C1M.LE.BSCLE) GO TO 70
      KFLAG = KFLAG + 1
      BSCLE = BRY(KFLAG)
      S1 = S1*CS
      S2 = ST
      S1 = S1*CSS(KFLAG)
```

```
        S2 = S2*CSS(KFLAG)
        CS = CSR(KFLAG)
   70 CONTINUE
      RETURN
   80 CONTINUE
      NZ = -1
      IF(NW.EQ.(-2)) NZ=-2
      RETURN
      END
      SUBROUTINE CAIRY(Z, ID, KODE, AI, NZ, IERR)
C***BEGIN PROLOGUE  CAIRY
C***DATE WRITTEN   830501   (YYMMDD)
C***REVISION DATE  890801   (YYMMDD)
C***CATEGORY NO.  B5K
C***KEYWORDS  AIRY FUNCTION,BESSEL FUNCTIONS OF ORDER ONE THIRD
C***AUTHOR  AMOS, DONALD E., SANDIA NATIONAL LABORATORIES
C***PURPOSE  TO COMPUTE AIRY FUNCTIONS AI(Z) AND DAI(Z) FOR COMPLEX Z
C***DESCRIPTION
C        ON KODE=1, CAIRY COMPUTES THE COMPLEX AIRY FUNCTION AI(Z) OR
C        ITS DERIVATIVE DAI(Z)/DZ ON ID=0 OR ID=1 RESPECTIVELY. ON
C        KODE=2, A SCALING OPTION CEXP(ZTA)*AI(Z) OR CEXP(ZTA)*
C        DAI(Z)/DZ IS PROVIDED TO REMOVE THE EXPONENTIAL DECAY IN
C        -PI/3.LT.ARG(Z).LT.PI/3 AND THE EXPONENTIAL GROWTH IN
C        PI/3.LT.ABS(ARG(Z)).LT.PI WHERE ZTA=(2/3)*Z*CSQRT(Z)
C
C        WHILE THE AIRY FUNCTIONS AI(Z) AND DAI(Z)/DZ ARE ANALYTIC IN
C        THE WHOLE Z PLANE, THE CORRESPONDING SCALED FUNCTIONS DEFINED
C        FOR KODE=2 HAVE A CUT ALONG THE NEGATIVE REAL AXIS.
C        DEFINITIONS AND NOTATION ARE FOUND IN THE NBS HANDBOOK OF
C        MATHEMATICAL FUNCTIONS (REF. 1).
C
C        INPUT
C          Z      - Z=CMPLX(X,Y)
C          ID     - ORDER OF DERIVATIVE, ID=0 OR ID=1
C          KODE   - A PARAMETER TO INDICATE THE SCALING OPTION
C                   KODE= 1  RETURNS
C                            AI=AI(Z)                ON ID=0 OR
C                            AI=DAI(Z)/DZ            ON ID=1
C                       = 2  RETURNS
C                            AI=CEXP(ZTA)*AI(Z)      ON ID=0 OR
C                            AI=CEXP(ZTA)*DAI(Z)/DZ  ON ID=1 WHERE
C                            ZTA=(2/3)*Z*CSQRT(Z)
C
C        OUTPUT
C          AI     - COMPLEX ANSWER DEPENDING ON THE CHOICES FOR ID AND
C                   KODE
C          NZ     - UNDERFLOW INDICATOR
C                   NZ= 0   , NORMAL RETURN
C                   NZ= 1   , AI=CMPLX(0.0,0.0) DUE TO UNDERFLOW IN
C                             -PI/3.LT.ARG(Z).LT.PI/3 ON KODE=1
C          IERR   - ERROR FLAG
C                   IERR=0, NORMAL RETURN - COMPUTATION COMPLETED
C                   IERR=1, INPUT ERROR   - NO COMPUTATION
C                   IERR=2, OVERFLOW      - NO COMPUTATION, REAL(ZTA)
C                           TOO LARGE WITH KODE=1.
C                   IERR=3, CABS(Z) LARGE      - COMPUTATION COMPLETED
C                           LOSSES OF SIGNIFCANCE BY ARGUMENT REDUCTION
C                           PRODUCE LESS THAN HALF OF MACHINE ACCURACY
C                   IERR=4, CABS(Z) TOO LARGE  - NO COMPUTATION
C                           COMPLETE LOSS OF ACCURACY BY ARGUMENT
C                           REDUCTION
```

```
C               IERR=5, ERROR            - NO COMPUTATION,
C                    ALGORITHM TERMINATION CONDITION NOT MET
C
C***LONG DESCRIPTION
C
C       AI AND DAI ARE COMPUTED FOR CABS(Z).GT.1.0 FROM THE K BESSEL
C       FUNCTIONS BY
C
C         AI(Z)=C*SQRT(Z)*K(1/3,ZTA) , DAI(Z)=-C*Z*K(2/3,ZTA)
C                    C=1.0/(PI*SQRT(3.0))
C                    ZTA=(2/3)*Z**(3/2)
C
C       WITH THE POWER SERIES FOR CABS(Z).LE.1.0.
C
C       IN MOST COMPLEX VARIABLE COMPUTATION, ONE MUST EVALUATE ELE-
C       MENTARY FUNCTIONS. WHEN THE MAGNITUDE OF Z IS LARGE, LOSSES
C       OF SIGNIFICANCE BY ARGUMENT REDUCTION OCCUR. CONSEQUENTLY, IF
C       THE MAGNITUDE OF ZETA=(2/3)*Z**1.5 EXCEEDS U1=SQRT(0.5/UR),
C       THEN LOSSES EXCEEDING HALF PRECISION ARE LIKELY AND AN ERROR
C       FLAG IERR=3 IS TRIGGERED WHERE UR=R1MACH(4)=UNIT ROUNDOFF.
C       ALSO, IF THE MAGNITUDE OF ZETA IS LARGER THAN U2=0.5/UR, THEN
C       ALL SIGNIFICANCE IS LOST AND IERR=4. IN ORDER TO USE THE INT
C       FUNCTION, ZETA MUST BE FURTHER RESTRICTED NOT TO EXCEED THE
C       LARGEST INTEGER, U3=I1MACH(9). THUS, THE MAGNITUDE OF ZETA
C       MUST BE RESTRICTED BY MIN(U2,U3). ON 32 BIT MACHINES, U1,U2,
C       AND U3 ARE APPROXIMATELY 2.0E+3, 4.2E+6, 2.1E+9 IN SINGLE
C       PRECISION ARITHMETIC AND 1.3E+8, 1.8E+16, 2.1E+9 IN DOUBLE
C       PRECISION ARITHMETIC RESPECTIVELY. THIS MAKES U2 AND U3 LIMIT-
C       ING IN THEIR RESPECTIVE ARITHMETICS. THIS MEANS THAT THE MAG-
C       NITUDE OF Z CANNOT EXCEED 3.1E+4 IN SINGLE AND 2.1E+6 IN
C       DOUBLE PRECISION ARITHMETIC. THIS ALSO MEANS THAT ONE CAN
C       EXPECT TO RETAIN, IN THE WORST CASES ON 32 BIT MACHINES,
C       NO DIGITS IN SINGLE PRECISION AND ONLY 7 DIGITS IN DOUBLE
C       PRECISION ARITHMETIC. SIMILAR CONSIDERATIONS HOLD FOR OTHER
C       MACHINES.
C
C       THE APPROXIMATE RELATIVE ERROR IN THE MAGNITUDE OF A COMPLEX
C       BESSEL FUNCTION CAN BE EXPRESSED BY P*10**S WHERE P=MAX(UNIT
C       ROUNDOFF,1.0E-18) IS THE NOMINAL PRECISION AND 10**S REPRE-
C       SENTS THE INCREASE IN ERROR DUE TO ARGUMENT REDUCTION IN THE
C       ELEMENTARY FUNCTIONS. HERE, S=MAX(1,ABS(LOG10(CABS(Z))),
C       ABS(LOG10(FNU))) APPROXIMATELY (I.E. S=MAX(1,ABS(EXPONENT OF
C       CABS(Z),ABS(EXPONENT OF FNU)) ). HOWEVER, THE PHASE ANGLE MAY
C       HAVE ONLY ABSOLUTE ACCURACY. THIS IS MOST LIKELY TO OCCUR WHEN
C       ONE COMPONENT (IN ABSOLUTE VALUE) IS LARGER THAN THE OTHER BY
C       SEVERAL ORDERS OF MAGNITUDE. IF ONE COMPONENT IS 10**K LARGER
C       THAN THE OTHER, THEN ONE CAN EXPECT ONLY MAX(ABS(LOG10(P))-K,
C       0) SIGNIFICANT DIGITS; OR, STATED ANOTHER WAY, WHEN K EXCEEDS
C       THE EXPONENT OF P, NO SIGNIFICANT DIGITS REMAIN IN THE SMALLER
C       COMPONENT. HOWEVER, THE PHASE ANGLE RETAINS ABSOLUTE
ACCURACY
C       BECAUSE, IN COMPLEX ARITHMETIC WITH PRECISION P, THE SMALLER
C       COMPONENT WILL NOT (AS A RULE) DECREASE BELOW P TIMES THE
C       MAGNITUDE OF THE LARGER COMPONENT. IN THESE EXTREME CASES,
C       THE PRINCIPAL PHASE ANGLE IS ON THE ORDER OF +P, -P, PI/2-P,
C       OR -PI/2+P.
C
C***REFERENCES  HANDBOOK OF MATHEMATICAL FUNCTIONS BY M. ABRAMOWITZ
C            AND I. A. STEGUN, NBS AMS SERIES 55, U.S. DEPT. OF
C            COMMERCE, 1955.
C
```

```
C           COMPUTATION OF BESSEL FUNCTIONS OF COMPLEX ARGUMENT
C             AND LARGE ORDER BY D. E. AMOS, SAND83-0643, MAY, 1983
C
C           A SUBROUTINE PACKAGE FOR BESSEL FUNCTIONS OF A COMPLEX
C             ARGUMENT AND NONNEGATIVE ORDER BY D. E. AMOS, SAND85-
C             1018, MAY, 1985
C
C           A PORTABLE PACKAGE FOR BESSEL FUNCTIONS OF A COMPLEX
C             ARGUMENT AND NONNEGATIVE ORDER BY D. E. AMOS, TRANS.
C             MATH. SOFTWARE, 1986
C***ROUTINES CALLED  CACAI,CBKNU,I1MACH,R1MACH
C***END PROLOGUE  CAIRY
      COMPLEX AI, CONE, CSQ, CY, S1, S2, TRM1, TRM2, Z, ZTA, Z3
      REAL AA, AD, AK, ALIM, ATRM, AZ, AZ3, BK, CK, COEF, C1, C2, DIG,
     * DK, D1, D2, ELIM, FID, FNU, RL, R1M5, SFAC, TOL, TTH, ZI, ZR,
     * Z3I, Z3R, R1MACH, BB, ALAZ
      INTEGER ID, IERR, IFLAG, K, KODE, K1, K2, MR, NN, NZ, I1MACH
      DIMENSION CY(1)
      DATA TTH, C1, C2, COEF /6.66666666666666667E-01,
     * 3.55028053887817240E-01,2.58819403792806799E-01,
     * 1.83776298473930683E-01/
      DATA  CONE / (1.0E0,0.0E0) /
C***FIRST EXECUTABLE STATEMENT  CAIRY
      IERR = 0
      NZ=0
      IF (ID.LT.0 .OR. ID.GT.1) IERR=1
      IF (KODE.LT.1 .OR. KODE.GT.2) IERR=1
      IF (IERR.NE.0) RETURN
      AZ = CABS(Z)
      TOL = AMAX1(R1MACH(4),1.0E-18)
      FID = FLOAT(ID)
      IF (AZ.GT.1.0E0) GO TO 60
C-----------------------------------------------------------------------
C     POWER SERIES FOR CABS(Z).LE.1.
C-----------------------------------------------------------------------
      S1 = CONE
      S2 = CONE
      IF (AZ.LT.TOL) GO TO 160
      AA = AZ*AZ
      IF (AA.LT.TOL/AZ) GO TO 40
      TRM1 = CONE
      TRM2 = CONE
      ATRM = 1.0E0
      Z3 = Z*Z*Z
      AZ3 = AZ*AA
      AK = 2.0E0 + FID
      BK = 3.0E0 - FID - FID
      CK = 4.0E0 - FID
      DK = 3.0E0 + FID + FID
      D1 = AK*DK
      D2 = BK*CK
      AD = AMIN1(D1,D2)
      AK = 24.0E0 + 9.0E0*FID
      BK = 30.0E0 - 9.0E0*FID
      Z3R = REAL(Z3)
      Z3I = AIMAG(Z3)
      DO 30 K=1,25
        TRM1 = TRM1*CMPLX(Z3R/D1,Z3I/D1)
        S1 = S1 + TRM1
        TRM2 = TRM2*CMPLX(Z3R/D2,Z3I/D2)
        S2 = S2 + TRM2
```

```
        ATRM = ATRM*AZ3/AD
        D1 = D1 + AK
        D2 = D2 + BK
        AD = AMIN1(D1,D2)
        IF (ATRM.LT.TOL*AD) GO TO 40
        AK = AK + 18.0E0
        BK = BK + 18.0E0
  30 CONTINUE
  40 CONTINUE
      IF (ID.EQ.1) GO TO 50
      AI = S1*CMPLX(C1,0.0E0) - Z*S2*CMPLX(C2,0.0E0)
      IF (KODE.EQ.1) RETURN
      ZTA = Z*CSQRT(Z)*CMPLX(TTH,0.0E0)
      AI = AI*CEXP(ZTA)
      RETURN
  50 CONTINUE
      AI = -S2*CMPLX(C2,0.0E0)
      IF (AZ.GT.TOL) AI = AI + Z*Z*S1*CMPLX(C1/(1.0E0+FID),0.0E0)
      IF (KODE.EQ.1) RETURN
      ZTA = Z*CSQRT(Z)*CMPLX(TTH,0.0E0)
      AI = AI*CEXP(ZTA)
      RETURN
C-----------------------------------------------------------------
C    CASE FOR CABS(Z).GT.1.0
C-----------------------------------------------------------------
  60 CONTINUE
      FNU = (1.0E0+FID)/3.0E0
C-----------------------------------------------------------------
C    SET PARAMETERS RELATED TO MACHINE CONSTANTS.
C    TOL IS THE APPROXIMATE UNIT ROUNDOFF LIMITED TO 1.0E-18.
C    ELIM IS THE APPROXIMATE EXPONENTIAL OVER- AND UNDERFLOW LIMIT.
C    EXP(-ELIM).LT.EXP(-ALIM)=EXP(-ELIM)/TOL    AND
C    EXP(ELIM).GT.EXP(ALIM)=EXP(ELIM)*TOL       ARE INTERVALS NEAR
C    UNDERFLOW AND OVERFLOW LIMITS WHERE SCALED ARITHMETIC IS DONE.
C    RL IS THE LOWER BOUNDARY OF THE ASYMPTOTIC EXPANSION FOR LARGE Z.
C    DIG = NUMBER OF BASE 10 DIGITS IN TOL = 10**(-DIG).
C-----------------------------------------------------------------
      K1 = I1MACH(12)
      K2 = I1MACH(13)
      R1M5 = R1MACH(5)
      K = MIN0(IABS(K1),IABS(K2))
      ELIM = 2.303E0*(FLOAT(K)*R1M5-3.0E0)
      K1 = I1MACH(11) - 1
      AA = R1M5*FLOAT(K1)
      DIG = AMIN1(AA,18.0E0)
      AA = AA*2.303E0
      ALIM = ELIM + AMAX1(-AA,-41.45E0)
      RL = 1.2E0*DIG + 3.0E0
      ALAZ=ALOG(AZ)
C-----------------------------------------------------------------
C    TEST FOR RANGE
C-----------------------------------------------------------------
      AA=0.5E0/TOL
      BB=FLOAT(I1MACH(9))*0.5E0
      AA=AMIN1(AA,BB)
      AA=AA**TTH
      IF (AZ.GT.AA) GO TO 260
      AA=SQRT(AA)
      IF (AZ.GT.AA) IERR=3
      CSQ=CSQRT(Z)
      ZTA=Z*CSQ*CMPLX(TTH,0.0E0)
```

```
C-----------------------------------------------------------------
C    RE(ZTA).LE.0 WHEN RE(Z).LT.0, ESPECIALLY WHEN IM(Z) IS SMALL
C-----------------------------------------------------------------
      IFLAG = 0
      SFAC = 1.0E0
      ZI = AIMAG(Z)
      ZR = REAL(Z)
      AK = AIMAG(ZTA)
      IF (ZR.GE.0.0E0) GO TO 70
      BK = REAL(ZTA)
      CK = -ABS(BK)
      ZTA = CMPLX(CK,AK)
   70 CONTINUE
      IF (ZI.NE.0.0E0) GO TO 80
      IF (ZR.GT.0.0E0) GO TO 80
      ZTA = CMPLX(0.0E0,AK)
   80 CONTINUE
      AA = REAL(ZTA)
      IF (AA.GE.0.0E0 .AND. ZR.GT.0.0E0) GO TO 100
      IF (KODE.EQ.2) GO TO 90
C-----------------------------------------------------------------
C    OVERFLOW TEST
C-----------------------------------------------------------------
      IF (AA.GT.(-ALIM)) GO TO 90
      AA = -AA + 0.25E0*ALAZ
      IFLAG = 1
      SFAC = TOL
      IF (AA.GT.ELIM) GO TO 240
   90 CONTINUE
C-----------------------------------------------------------------
C    CBKNU AND CACAI RETURN EXP(ZTA)*K(FNU,ZTA) ON KODE=2
C-----------------------------------------------------------------
      MR = 1
      IF (ZI.LT.0.0E0) MR = -1
      CALL CACAI(ZTA, FNU, KODE, MR, 1, CY, NN, RL, TOL, ELIM, ALIM)
      IF (NN.LT.0) GO TO 250
      NZ = NZ + NN
      GO TO 120
  100 CONTINUE
      IF (KODE.EQ.2) GO TO 110
C-----------------------------------------------------------------
C    UNDERFLOW TEST
C-----------------------------------------------------------------
      IF (AA.LT.ALIM) GO TO 110
      AA = -AA - 0.25E0*ALAZ
      IFLAG = 2
      SFAC = 1.0E0/TOL
      IF (AA.LT.(-ELIM)) GO TO 180
  110 CONTINUE
      CALL CBKNU(ZTA, FNU, KODE, 1, CY, NZ, TOL, ELIM, ALIM)
  120 CONTINUE
      S1 = CY(1)*CMPLX(COEF,0.0E0)
      IF (IFLAG.NE.0) GO TO 140
      IF (ID.EQ.1) GO TO 130
      AI = CSQ*S1
      RETURN
  130 AI = -Z*S1
      RETURN
  140 CONTINUE
      S1 = S1*CMPLX(SFAC,0.0E0)
      IF (ID.EQ.1) GO TO 150
```

```
      S1 = S1*CSQ
      AI = S1*CMPLX(1.0E0/SFAC,0.0E0)
      RETURN
  150 CONTINUE
      S1 = -S1*Z
      AI = S1*CMPLX(1.0E0/SFAC,0.0E0)
      RETURN
  160 CONTINUE
      AA = 1.0E+3*R1MACH(1)
      S1 = CMPLX(0.0E0,0.0E0)
      IF (ID.EQ.1) GO TO 170
      IF (AZ.GT.AA) S1 = CMPLX(C2,0.0E0)*Z
      AI = CMPLX(C1,0.0E0) - S1
      RETURN
  170 CONTINUE
      AI = -CMPLX(C2,0.0E0)
      AA = SQRT(AA)
      IF (AZ.GT.AA) S1 = Z*Z*CMPLX(0.5E0,0.0E0)
      AI = AI + S1*CMPLX(C1,0.0E0)
      RETURN
  180 CONTINUE
      NZ = 1
      AI = CMPLX(0.0E0,0.0E0)
      RETURN
  240 CONTINUE
      NZ = 0
      IERR=2
      RETURN
  250 CONTINUE
      IF(NN.EQ.(-1)) GO TO 240
      NZ=0
      IERR=5
      RETURN
  260 CONTINUE
      IERR=4
      NZ=0
      RETURN
      END
      SUBROUTINE CASYI(Z, FNU, KODE, N, Y, NZ, RL, TOL, ELIM, ALIM)
C***BEGIN PROLOGUE  CASYI
C***REFER TO  CBESI,CBESK
C
C     CASYI COMPUTES THE I BESSEL FUNCTION FOR REAL(Z).GE.0.0 BY
C     MEANS OF THE ASYMPTOTIC EXPANSION FOR LARGE CABS(Z) IN THE
C     REGION CABS(Z).GT.MAX(RL,FNU*FNU/2). NZ=0 IS A NORMAL RETURN.
C     NZ.LT.0 INDICATES AN OVERFLOW ON KODE=1.
C
C***ROUTINES CALLED  R1MACH
C***END PROLOGUE  CASYI
      COMPLEX AK1, CK, CONE, CS1, CS2, CZ, CZERO, DK, EZ, P1, RZ, S2,
     * Y, Z
      REAL AA, ACZ, AEZ, AK, ALIM, ARG, ARM, ATOL, AZ, BB, BK, DFNU,
     * DNU2, ELIM, FDN, FNU, PI, RL, RTPI, RTR1, S, SGN, SQK, TOL, X,
     * YY, R1MACH
      INTEGER I, IB, IL, INU, J, JL, K, KODE, KODED, M, N, NN, NZ
      DIMENSION Y(N)
      DATA PI, RTPI  /3.14159265358979324E0 , 0.159154943091895336E0 /
      DATA CZERO, CONE / (0.0E0,0.0E0), (1.0E0,0.0E0) /
      NZ = 0
      AZ = CABS(Z)
      X = REAL(Z)
```

```
      ARM = 1.0E+3*R1MACH(1)
      RTR1 = SQRT(ARM)
      IL = MIN0(2,N)
      DFNU = FNU + FLOAT(N-IL)
C-----------------------------------------------------------------------
C     OVERFLOW TEST
C-----------------------------------------------------------------------
      AK1 = CMPLX(RTPI,0.0E0)/Z
      AK1 = CSQRT(AK1)
      CZ = Z
      IF (KODE.EQ.2) CZ = Z - CMPLX(X,0.0E0)
      ACZ = REAL(CZ)
      IF (ABS(ACZ).GT.ELIM) GO TO 80
      DNU2 = DFNU + DFNU
      KODED = 1
      IF ((ABS(ACZ).GT.ALIM) .AND. (N.GT.2)) GO TO 10
      KODED = 0
      AK1 = AK1*CEXP(CZ)
   10 CONTINUE
      FDN = 0.0E0
      IF (DNU2.GT.RTR1) FDN = DNU2*DNU2
      EZ = Z*CMPLX(8.0E0,0.0E0)
C-----------------------------------------------------------------------
C     WHEN Z IS IMAGINARY, THE ERROR TEST MUST BE MADE RELATIVE TO THE
C     FIRST RECIPROCAL POWER SINCE THIS IS THE LEADING TERM OF THE
C     EXPANSION FOR THE IMAGINARY PART.
C-----------------------------------------------------------------------
      AEZ = 8.0E0*AZ
      S = TOL/AEZ
      JL = INT(RL+RL) + 2
      YY = AIMAG(Z)
      P1 = CZERO
      IF (YY.EQ.0.0E0) GO TO 20
C-----------------------------------------------------------------------
C     CALCULATE EXP(PI*(0.5+FNU+N-IL)*I) TO MINIMIZE LOSSES OF
C     SIGNIFICANCE WHEN FNU OR N IS LARGE
C-----------------------------------------------------------------------
      INU = INT(FNU)
      ARG = (FNU-FLOAT(INU))*PI
      INU = INU + N - IL
      AK = -SIN(ARG)
      BK = COS(ARG)
      IF (YY.LT.0.0E0) BK = -BK
      P1 = CMPLX(AK,BK)
      IF (MOD(INU,2).EQ.1) P1 = -P1
   20 CONTINUE
      DO 50 K=1,IL
        SQK = FDN - 1.0E0
        ATOL = S*ABS(SQK)
        SGN = 1.0E0
        CS1 = CONE
        CS2 = CONE
        CK = CONE
        AK = 0.0E0
        AA = 1.0E0
        BB = AEZ
        DK = EZ
        DO 30 J=1,JL
          CK = CK*CMPLX(SQK,0.0E0)/DK
          CS2 = CS2 + CK
          SGN = -SGN
```

```
     CS1 = CS1 + CK*CMPLX(SGN,0.0E0)
     DK = DK + EZ
     AA = AA*ABS(SQK)/BB
     BB = BB + AEZ
     AK = AK + 8.0E0
     SQK = SQK - AK
     IF (AA.LE.ATOL) GO TO 40
 30  CONTINUE
    GO TO 90
 40  CONTINUE
    S2 = CS1
    IF (X+X.LT.ELIM) S2 = S2 + P1*CS2*CEXP(-Z-Z)
    FDN = FDN + 8.0E0*DFNU + 4.0E0
    P1 = -P1
    M = N - IL + K
    Y(M) = S2*AK1
 50 CONTINUE
   IF (N.LE.2) RETURN
   NN = N
   K = NN - 2
   AK = FLOAT(K)
   RZ = (CONE+CONE)/Z
   IB = 3
   DO 60 I=IB,NN
    Y(K) = CMPLX(AK+FNU,0.0E0)*RZ*Y(K+1) + Y(K+2)
    AK = AK - 1.0E0
    K = K - 1
 60 CONTINUE
   IF (KODED.EQ.0) RETURN
   CK = CEXP(CZ)
   DO 70 I=1,NN
    Y(I) = Y(I)*CK
 70 CONTINUE
   RETURN
 80 CONTINUE
   NZ = -1
   RETURN
 90 CONTINUE
   NZ=-2
   RETURN
   END
   SUBROUTINE CBINU(Z, FNU, KODE, N, CY, NZ, RL, FNUL, TOL, ELIM,
  * ALIM)
C***BEGIN PROLOGUE  CBINU
C***REFER TO  CBESH,CBESI,CBESJ,CBESK,CAIRY,CBIRY
C
C    CBINU COMPUTES THE I FUNCTION IN THE RIGHT HALF Z PLANE
C
C***ROUTINES CALLED  CASYI,CBUNI,CMLRI,CSERI,CUOIK,CWRSK
C***END PROLOGUE  CBINU
   COMPLEX CW, CY, CZERO, Z
   REAL ALIM, AZ, DFNU, ELIM, FNU, FNUL, RL, TOL
   INTEGER I, INW, KODE, N, NLAST, NN, NUI, NW, NZ
   DIMENSION CY(N), CW(2)
   DATA CZERO / (0.0E0,0.0E0) /
C
   NZ = 0
   AZ = CABS(Z)
   NN = N
   DFNU = FNU + FLOAT(N-1)
   IF (AZ.LE.2.0E0) GO TO 10
```

```
      IF (AZ*AZ*0.25E0.GT.DFNU+1.0E0) GO TO 20
   10 CONTINUE
C-----------------------------------------------------------------------
C     POWER SERIES
C-----------------------------------------------------------------------
      CALL CSERI(Z, FNU, KODE, NN, CY, NW, TOL, ELIM, ALIM)
      INW = IABS(NW)
      NZ = NZ + INW
      NN = NN - INW
      IF (NN.EQ.0) RETURN
      IF (NW.GE.0) GO TO 120
      DFNU = FNU + FLOAT(NN-1)
   20 CONTINUE
      IF (AZ.LT.RL) GO TO 40
      IF (DFNU.LE.1.0E0) GO TO 30
      IF (AZ+AZ.LT.DFNU*DFNU) GO TO 50
C-----------------------------------------------------------------------
C     ASYMPTOTIC EXPANSION FOR LARGE Z
C-----------------------------------------------------------------------
   30 CONTINUE
      CALL CASYI(Z, FNU, KODE, NN, CY, NW, RL, TOL, ELIM, ALIM)
      IF (NW.LT.0) GO TO 130
      GO TO 120
   40 CONTINUE
      IF (DFNU.LE.1.0E0) GO TO 70
   50 CONTINUE
C-----------------------------------------------------------------------
C     OVERFLOW AND UNDERFLOW TEST ON I SEQUENCE FOR MILLER ALGORITHM
C-----------------------------------------------------------------------
      CALL CUOIK(Z, FNU, KODE, 1, NN, CY, NW, TOL, ELIM, ALIM)
      IF (NW.LT.0) GO TO 130
      NZ = NZ + NW
      NN = NN - NW
      IF (NN.EQ.0) RETURN
      DFNU = FNU+FLOAT(NN-1)
      IF (DFNU.GT.FNUL) GO TO 110
      IF (AZ.GT.FNUL) GO TO 110
   60 CONTINUE
      IF (AZ.GT.RL) GO TO 80
   70 CONTINUE
C-----------------------------------------------------------------------
C     MILLER ALGORITHM NORMALIZED BY THE SERIES
C-----------------------------------------------------------------------
      CALL CMLRI(Z, FNU, KODE, NN, CY, NW, TOL)
      IF(NW.LT.0) GO TO 130
      GO TO 120
   80 CONTINUE
C-----------------------------------------------------------------------
C     MILLER ALGORITHM NORMALIZED BY THE WRONSKIAN
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C     OVERFLOW TEST ON K FUNCTIONS USED IN WRONSKIAN
C-----------------------------------------------------------------------
      CALL CUOIK(Z, FNU, KODE, 2, 2, CW, NW, TOL, ELIM, ALIM)
      IF (NW.GE.0) GO TO 100
      NZ = NN
      DO 90 I=1,NN
        CY(I) = CZERO
   90 CONTINUE
      RETURN
  100 CONTINUE
```

```
      IF (NW.GT.0) GO TO 130
      CALL CWRSK(Z, FNU, KODE, NN, CY, NW, CW, TOL, ELIM, ALIM)
      IF (NW.LT.0) GO TO 130
      GO TO 120
  110 CONTINUE
C-----------------------------------------------------------------------
C     INCREMENT FNU+NN-1 UP TO FNUL, COMPUTE AND RECUR BACKWARD
C-----------------------------------------------------------------------
      NUI = INT(FNUL-DFNU) + 1
      NUI = MAX0(NUI,0)
      CALL CBUNI(Z, FNU, KODE, NN, CY, NW, NUI, NLAST, FNUL, TOL, ELIM,
     * ALIM)
      IF (NW.LT.0) GO TO 130
      NZ = NZ + NW
      IF (NLAST.EQ.0) GO TO 120
      NN = NLAST
      GO TO 60
  120 CONTINUE
      RETURN
  130 CONTINUE
      NZ = -1
      IF(NW.EQ.(-2)) NZ=-2
      RETURN
      END
      SUBROUTINE CBKNU(Z, FNU, KODE, N, Y, NZ, TOL, ELIM, ALIM)
C***BEGIN PROLOGUE  CBKNU
C***REFER TO  CBESI,CBESK,CAIRY,CBESH
C
C     CBKNU COMPUTES THE K BESSEL FUNCTION IN THE RIGHT HALF Z PLANE
C
C***ROUTINES CALLED  CKSCL,CSHCH,GAMLN,I1MACH,R1MACH,CUCHK
C***END PROLOGUE  CBKNU
C
      COMPLEX CCH, CK, COEF, CONE, CRSC, CS, CSCL, CSH, CSR, CSS, CTWO,
     * CZ, CZERO, F, FMU, P, PT, P1, P2, Q, RZ, SMU, ST, S1, S2, Y, Z,
     * ZD, CELM, CY
      REAL AA, AK, ALIM, ASCLE, A1, A2, BB, BK, BRY, CAZ, CC, DNU,
     * DNU2, ELIM, ETEST, FC, FHS, FK, FKS, FNU, FPI, G1, G2, HPI, PI,
     * P2I, P2M, P2R, RK, RTHPI, R1, S, SPI, TM, TOL, TTH, T1, T2, XX,
     * YY, GAMLN, R1MACH, HELIM, ELM, XD, YD, ALAS, AS
      INTEGER I, IDUM, IFLAG, INU, K, KFLAG, KK, KMAX, KODE, KODED, N,
     * NZ, I1MACH, NW, J, IC, INUB
      DIMENSION BRY(3), CC(8), CSS(3), CSR(3), Y(N), CY(2)
      DATA KMAX / 30 /
      DATA R1 / 2.0E0 /
      DATA CZERO,CONE,CTWO /(0.0E0,0.0E0),(1.0E0,0.0E0),(2.0E0,0.0E0)/
      DATA PI, RTHPI, SPI ,HPI, FPI, TTH /
     1     3.14159265358979324E0,       1.25331413731550025E0,
     2     1.90985931710274403E0,       1.57079632679489662E0,
     3     1.89769999331517738E0,       6.66666666666666666E-01/
      DATA CC(1), CC(2), CC(3), CC(4), CC(5), CC(6), CC(7), CC(8)/
     1     5.77215664901532861E-01,    -4.20026350340952355E-02,
     2    -4.21977345555443367E-02,     7.21894324666309954E-03,
     3    -2.15241674114950973E-04,    -2.01348547807882387E-05,
     4     1.13302723198169588E-06,     6.11609510448141582E-09/
      XX = REAL(Z)
      YY = AIMAG(Z)
      CAZ = CABS(Z)
      CSCL = CMPLX(1.0E0/TOL,0.0E0)
      CRSC = CMPLX(TOL,0.0E0)
      CSS(1) = CSCL
```

```
      CSS(2) = CONE
      CSS(3) = CRSC
      CSR(1) = CRSC
      CSR(2) = CONE
      CSR(3) = CSCL
      BRY(1) = 1.0E+3*R1MACH(1)/TOL
      BRY(2) = 1.0E0/BRY(1)
      BRY(3) = R1MACH(2)
      NZ = 0
      IFLAG = 0
      KODED = KODE
      RZ = CTWO/Z
      INU = INT(FNU+0.5E0)
      DNU = FNU - FLOAT(INU)
      IF (ABS(DNU).EQ.0.5E0) GO TO 110
      DNU2 = 0.0E0
      IF (ABS(DNU).GT.TOL) DNU2 = DNU*DNU
      IF (CAZ.GT.R1) GO TO 110
C-----------------------------------------------------------------------
C     SERIES FOR CABS(Z).LE.R1
C-----------------------------------------------------------------------
      FC = 1.0E0
      SMU = CLOG(RZ)
      FMU = SMU*CMPLX(DNU,0.0E0)
      CALL CSHCH(FMU, CSH, CCH)
      IF (DNU.EQ.0.0E0) GO TO 10
      FC = DNU*PI
      FC = FC/SIN(FC)
      SMU = CSH*CMPLX(1.0E0/DNU,0.0E0)
   10 CONTINUE
      A2 = 1.0E0 + DNU
C-----------------------------------------------------------------------
C     GAM(1-Z)*GAM(1+Z)=PI*Z/SIN(PI*Z), T1=1/GAM(1-DNU), T2=1/GAM(1+DNU)
C-----------------------------------------------------------------------
      T2 = EXP(-GAMLN(A2,IDUM))
      T1 = 1.0E0/(T2*FC)
      IF (ABS(DNU).GT.0.1E0) GO TO 40
C-----------------------------------------------------------------------
C     SERIES FOR F0 TO RESOLVE INDETERMINACY FOR SMALL ABS(DNU)
C-----------------------------------------------------------------------
      AK = 1.0E0
      S = CC(1)
      DO 20 K=2,8
        AK = AK*DNU2
        TM = CC(K)*AK
        S = S + TM
        IF (ABS(TM).LT.TOL) GO TO 30
   20 CONTINUE
   30 G1 = -S
      GO TO 50
   40 CONTINUE
      G1 = (T1-T2)/(DNU+DNU)
   50 CONTINUE
      G2 = 0.5E0*(T1+T2)*FC
      G1 = G1*FC
      F = CMPLX(G1,0.0E0)*CCH + SMU*CMPLX(G2,0.0E0)
      PT = CEXP(FMU)
      P = CMPLX(0.5E0/T2,0.0E0)*PT
      Q = CMPLX(0.5E0/T1,0.0E0)/PT
      S1 = F
      S2 = P
```

```
      AK = 1.0E0
      A1 = 1.0E0
      CK = CONE
      BK = 1.0E0 - DNU2
      IF (INU.GT.0 .OR. N.GT.1) GO TO 80
C-----------------------------------------------------------------------
C     GENERATE K(FNU,Z), 0.0D0 .LE. FNU .LT. 0.5D0 AND N=1
C-----------------------------------------------------------------------
      IF (CAZ.LT.TOL) GO TO 70
      CZ = Z*Z*CMPLX(0.25E0,0.0E0)
      T1 = 0.25E0*CAZ*CAZ
   60 CONTINUE
      F = (F*CMPLX(AK,0.0E0)+P+Q)*CMPLX(1.0E0/BK,0.0E0)
      P = P*CMPLX(1.0E0/(AK-DNU),0.0E0)
      Q = Q*CMPLX(1.0E0/(AK+DNU),0.0E0)
      RK = 1.0E0/AK
      CK = CK*CZ*CMPLX(RK,0.0)
      S1 = S1 + CK*F
      A1 = A1*T1*RK
      BK = BK + AK + AK + 1.0E0
      AK = AK + 1.0E0
      IF (A1.GT.TOL) GO TO 60
   70 CONTINUE
      Y(1) = S1
      IF (KODED.EQ.1) RETURN
      Y(1) = S1*CEXP(Z)
      RETURN
C-----------------------------------------------------------------------
C     GENERATE K(DNU,Z) AND K(DNU+1,Z) FOR FORWARD RECURRENCE
C-----------------------------------------------------------------------
   80 CONTINUE
      IF (CAZ.LT.TOL) GO TO 100
      CZ = Z*Z*CMPLX(0.25E0,0.0E0)
      T1 = 0.25E0*CAZ*CAZ
   90 CONTINUE
      F = (F*CMPLX(AK,0.0E0)+P+Q)*CMPLX(1.0E0/BK,0.0E0)
      P = P*CMPLX(1.0E0/(AK-DNU),0.0E0)
      Q = Q*CMPLX(1.0E0/(AK+DNU),0.0E0)
      RK = 1.0E0/AK
      CK = CK*CZ*CMPLX(RK,0.0E0)
      S1 = S1 + CK*F
      S2 = S2 + CK*(P-F*CMPLX(AK,0.0E0))
      A1 = A1*T1*RK
      BK = BK + AK + AK + 1.0E0
      AK = AK + 1.0E0
      IF (A1.GT.TOL) GO TO 90
  100 CONTINUE
      KFLAG = 2
      BK = REAL(SMU)
      A1 = FNU + 1.0E0
      AK = A1*ABS(BK)
      IF (AK.GT.ALIM) KFLAG = 3
      P2 = S2*CSS(KFLAG)
      S2 = P2*RZ
      S1 = S1*CSS(KFLAG)
      IF (KODED.EQ.1) GO TO 210
      F = CEXP(Z)
      S1 = S1*F
      S2 = S2*F
      GO TO 210
C-----------------------------------------------------------------------
```

```
C    IFLAG=0 MEANS NO UNDERFLOW OCCURRED
C    IFLAG=1 MEANS AN UNDERFLOW OCCURRED- COMPUTATION PROCEEDS
WITH
C    KODED=2 AND A TEST FOR ON SCALE VALUES IS MADE DURING FORWARD
C    RECURSION
C-----------------------------------------------------------------------
  110 CONTINUE
      COEF = CMPLX(RTHPI,0.0E0)/CSQRT(Z)
      KFLAG = 2
      IF (KODED.EQ.2) GO TO 120
      IF (XX.GT.ALIM) GO TO 290
C     BLANK LINE
      A1 = EXP(-XX)*REAL(CSS(KFLAG))
      PT = CMPLX(A1,0.0E0)*CMPLX(COS(YY),-SIN(YY))
      COEF = COEF*PT
  120 CONTINUE
      IF (ABS(DNU).EQ.0.5E0) GO TO 300
C-----------------------------------------------------------------------
C     MILLER ALGORITHM FOR CABS(Z).GT.R1
C-----------------------------------------------------------------------
      AK = COS(PI*DNU)
      AK = ABS(AK)
      IF (AK.EQ.0.0E0) GO TO 300
      FHS = ABS(0.25E0-DNU2)
      IF (FHS.EQ.0.0E0) GO TO 300
C-----------------------------------------------------------------------
C     COMPUTE R2=F(E). IF CABS(Z).GE.R2, USE FORWARD RECURRENCE TO
C     DETERMINE THE BACKWARD INDEX K. R2=F(E) IS A STRAIGHT LINE ON
C     12.LE.E.LE.60. E IS COMPUTED FROM 2**(-E)=B**(1-I1MACH(11))=
C     TOL WHERE B IS THE BASE OF THE ARITHMETIC.
C-----------------------------------------------------------------------
      T1 = FLOAT(I1MACH(11)-1)*R1MACH(5)*3.321928094E0
      T1 = AMAX1(T1,12.0E0)
      T1 = AMIN1(T1,60.0E0)
      T2 = TTH*T1 - 6.0E0
      IF (XX.NE.0.0E0) GO TO 130
      T1 = HPI
      GO TO 140
  130 CONTINUE
      T1 = ATAN(YY/XX)
      T1 = ABS(T1)
  140 CONTINUE
      IF (T2.GT.CAZ) GO TO 170
C-----------------------------------------------------------------------
C     FORWARD RECURRENCE LOOP WHEN CABS(Z).GE.R2
C-----------------------------------------------------------------------
      ETEST = AK/(PI*CAZ*TOL)
      FK = 1.0E0
      IF (ETEST.LT.1.0E0) GO TO 180
      FKS = 2.0E0
      RK = CAZ + CAZ + 2.0E0
      A1 = 0.0E0
      A2 = 1.0E0
      DO 150 I=1,KMAX
        AK = FHS/FKS
        BK = RK/(FK+1.0E0)
        TM = A2
        A2 = BK*A2 - AK*A1
        A1 = TM
        RK = RK + 2.0E0
        FKS = FKS + FK + FK + 2.0E0
```

```
        FHS = FHS + FK + FK
        FK = FK + 1.0E0
        TM = ABS(A2)*FK
        IF (ETEST.LT.TM) GO TO 160
  150 CONTINUE
      GO TO 310
  160 CONTINUE
      FK = FK + SPI*T1*SQRT(T2/CAZ)
      FHS = ABS(0.25E0-DNU2)
      GO TO 180
  170 CONTINUE
C-----------------------------------------------------------------------
C     COMPUTE BACKWARD INDEX K FOR CABS(Z).LT.R2
C-----------------------------------------------------------------------
      A2 = SQRT(CAZ)
      AK = FPI*AK/(TOL*SQRT(A2))
      AA = 3.0E0*T1/(1.0E0+CAZ)
      BB = 14.7E0*T1/(28.0E0+CAZ)
      AK = (ALOG(AK)+CAZ*COS(AA)/(1.0E0+0.008E0*CAZ))/COS(BB)
      FK = 0.12125E0*AK*AK/CAZ + 1.5E0
  180 CONTINUE
      K = INT(FK)
C-----------------------------------------------------------------------
C     BACKWARD RECURRENCE LOOP FOR MILLER ALGORITHM
C-----------------------------------------------------------------------
      FK = FLOAT(K)
      FKS = FK*FK
      P1 = CZERO
      P2 = CMPLX(TOL,0.0E0)
      CS = P2
      DO 190 I=1,K
        A1 = FKS - FK
        A2 = (FKS+FK)/(A1+FHS)
        RK = 2.0E0/(FK+1.0E0)
        T1 = (FK+XX)*RK
        T2 = YY*RK
        PT = P2
        P2 = (P2*CMPLX(T1,T2)-P1)*CMPLX(A2,0.0E0)
        P1 = PT
        CS = CS + P2
        FKS = A1 - FK + 1.0E0
        FK = FK - 1.0E0
  190 CONTINUE
C-----------------------------------------------------------------------
C     COMPUTE (P2/CS)=(P2/CABS(CS))*(CONJG(CS)/CABS(CS)) FOR BETTER
C     SCALING
C-----------------------------------------------------------------------
      TM = CABS(CS)
      PT = CMPLX(1.0E0/TM,0.0E0)
      S1 = PT*P2
      CS = CONJG(CS)*PT
      S1 = COEF*S1*CS
      IF (INU.GT.0 .OR. N.GT.1) GO TO 200
      ZD = Z
      IF(IFLAG.EQ.1) GO TO 270
      GO TO 240
  200 CONTINUE
C-----------------------------------------------------------------------
C     COMPUTE P1/P2=(P1/CABS(P2)*CONJG(P2)/CABS(P2) FOR SCALING
C-----------------------------------------------------------------------
      TM = CABS(P2)
```

```
      PT = CMPLX(1.0E0/TM,0.0E0)
      P1 = PT*P1
      P2 = CONJG(P2)*PT
      PT = P1*P2
      S2 = S1*(CONE+(CMPLX(DNU+0.5E0,0.0E0)-PT)/Z)
C-----------------------------------------------------------------
C    FORWARD RECURSION ON THE THREE TERM RECURSION RELATION WITH
C    SCALING NEAR EXPONENT EXTREMES ON KFLAG=1 OR KFLAG=3
C-----------------------------------------------------------------
  210 CONTINUE
      CK = CMPLX(DNU+1.0E0,0.0E0)*RZ
      IF (N.EQ.1) INU = INU - 1
      IF (INU.GT.0) GO TO 220
      IF (N.EQ.1) S1=S2
      ZD = Z
      IF(IFLAG.EQ.1) GO TO 270
      GO TO 240
  220 CONTINUE
      INUB = 1
      IF (IFLAG.EQ.1) GO TO 261
  225 CONTINUE
      P1 = CSR(KFLAG)
      ASCLE = BRY(KFLAG)
      DO 230 I=INUB,INU
        ST = S2
        S2 = CK*S2 + S1
        S1 = ST
        CK = CK + RZ
        IF (KFLAG.GE.3) GO TO 230
        P2 = S2*P1
        P2R = REAL(P2)
        P2I = AIMAG(P2)
        P2R = ABS(P2R)
        P2I = ABS(P2I)
        P2M = AMAX1(P2R,P2I)
        IF (P2M.LE.ASCLE) GO TO 230
        KFLAG = KFLAG + 1
        ASCLE = BRY(KFLAG)
        S1 = S1*P1
        S2 = P2
        S1 = S1*CSS(KFLAG)
        S2 = S2*CSS(KFLAG)
        P1 = CSR(KFLAG)
  230 CONTINUE
      IF (N.EQ.1) S1 = S2
  240 CONTINUE
      Y(1) = S1*CSR(KFLAG)
      IF (N.EQ.1) RETURN
      Y(2) = S2*CSR(KFLAG)
      IF (N.EQ.2) RETURN
      KK = 2
  250 CONTINUE
      KK = KK + 1
      IF (KK.GT.N) RETURN
      P1 = CSR(KFLAG)
      ASCLE = BRY(KFLAG)
      DO 260 I=KK,N
        P2 = S2
        S2 = CK*S2 + S1
        S1 = P2
        CK = CK + RZ
```

```
     P2 = S2*P1
     Y(I) = P2
     IF (KFLAG.GE.3) GO TO 260
     P2R = REAL(P2)
     P2I = AIMAG(P2)
     P2R = ABS(P2R)
     P2I = ABS(P2I)
     P2M = AMAX1(P2R,P2I)
     IF (P2M.LE.ASCLE) GO TO 260
     KFLAG = KFLAG + 1
     ASCLE = BRY(KFLAG)
     S1 = S1*P1
     S2 = P2
     S1 = S1*CSS(KFLAG)
     S2 = S2*CSS(KFLAG)
     P1 = CSR(KFLAG)
 260 CONTINUE
    RETURN
C-----------------------------------------------------------------
C    IFLAG=1 CASES, FORWARD RECURRENCE ON SCALED VALUES ON
UNDERFLOW
C-----------------------------------------------------------------
 261 CONTINUE
    HELIM = 0.5E0*ELIM
    ELM = EXP(-ELIM)
    CELM = CMPLX(ELM,0.0)
    ASCLE = BRY(1)
    ZD = Z
    XD = XX
    YD = YY
    IC = -1
    J = 2
    DO 262 I=1,INU
     ST = S2
     S2 = CK*S2+S1
     S1 = ST
     CK = CK+RZ
     AS = CABS(S2)
     ALAS = ALOG(AS)
     P2R = -XD+ALAS
     IF(P2R.LT.(-ELIM)) GO TO 263
     P2 = -ZD+CLOG(S2)
     P2R = REAL(P2)
     P2I = AIMAG(P2)
     P2M = EXP(P2R)/TOL
     P1 = CMPLX(P2M,0.0E0)*CMPLX(COS(P2I),SIN(P2I))
     CALL CUCHK(P1,NW,ASCLE,TOL)
     IF(NW.NE.0) GO TO 263
     J=3-J
     CY(J) = P1
     IF(IC.EQ.(I-1)) GO TO 264
     IC = I
     GO TO 262
 263   CONTINUE
     IF(ALAS.LT.HELIM) GO TO 262
     XD = XD-ELIM
     S1 = S1*CELM
     S2 = S2*CELM
     ZD = CMPLX(XD,YD)
 262 CONTINUE
   IF(N.EQ.1) S1 = S2
```

```
    GO TO 270
  264 CONTINUE
    KFLAG = 1
    INUB = I+1
    S2 = CY(J)
    J = 3 - J
    S1 = CY(J)
    IF(INUB.LE.INU) GO TO 225
    IF(N.EQ.1) S1 = S2
    GO TO 240
  270 CONTINUE
    Y(1) = S1
    IF (N.EQ.1) GO TO 280
    Y(2) = S2
  280 CONTINUE
    ASCLE = BRY(1)
    CALL CKSCL(ZD, FNU, N, Y, NZ, RZ, ASCLE, TOL, ELIM)
    INU = N - NZ
    IF (INU.LE.0) RETURN
    KK = NZ + 1
    S1 = Y(KK)
    Y(KK) = S1*CSR(1)
    IF (INU.EQ.1) RETURN
    KK = NZ + 2
    S2 = Y(KK)
    Y(KK) = S2*CSR(1)
    IF (INU.EQ.2) RETURN
    T2 = FNU + FLOAT(KK-1)
    CK = CMPLX(T2,0.0E0)*RZ
    KFLAG = 1
    GO TO 250
  290 CONTINUE
C-----------------------------------------------------------------
C    SCALE BY EXP(Z), IFLAG = 1 CASES
C-----------------------------------------------------------------
    KODED = 2
    IFLAG = 1
    KFLAG = 2
    GO TO 120
C-----------------------------------------------------------------
C    FNU=HALF ODD INTEGER CASE, DNU=-0.5
C-----------------------------------------------------------------
  300 CONTINUE
    S1 = COEF
    S2 = COEF
    GO TO 210
  310 CONTINUE
    NZ=-2
    RETURN
    END
    SUBROUTINE CBUNI(Z, FNU, KODE, N, Y, NZ, NUI, NLAST, FNUL, TOL,
   * ELIM, ALIM)
C***BEGIN PROLOGUE  CBUNI
C***REFER TO  CBESI,CBESK
C
C    CBUNI COMPUTES THE I BESSEL FUNCTION FOR LARGE CABS(Z).GT.
C    FNUL AND FNU+N-1.LT.FNUL. THE ORDER IS INCREASED FROM
C    FNU+N-1 GREATER THAN FNUL BY ADDING NUI AND COMPUTING
C    ACCORDING TO THE UNIFORM ASYMPTOTIC EXPANSION FOR I(FNU,Z)
C    ON IFORM=1 AND THE EXPANSION FOR J(FNU,Z) ON IFORM=2
C
```

```
C***ROUTINES CALLED  CUNI1,CUNI2,R1MACH
C***END PROLOGUE  CBUNI
      COMPLEX CSCL, CSCR, CY, RZ, ST, S1, S2, Y, Z
      REAL ALIM, AX, AY, DFNU, ELIM, FNU, FNUI, FNUL, GNU, TOL, XX, YY,
     * ASCLE, BRY, STR, STI, STM, R1MACH
      INTEGER I, IFLAG, IFORM, K, KODE, N, NL, NLAST, NUI, NW, NZ
      DIMENSION Y(N), CY(2), BRY(3)
      NZ = 0
      XX = REAL(Z)
      YY = AIMAG(Z)
      AX = ABS(XX)*1.7321E0
      AY = ABS(YY)
      IFORM = 1
      IF (AY.GT.AX) IFORM = 2
      IF (NUI.EQ.0) GO TO 60
      FNUI = FLOAT(NUI)
      DFNU = FNU + FLOAT(N-1)
      GNU = DFNU + FNUI
      IF (IFORM.EQ.2) GO TO 10
C-----------------------------------------------------------------------
C     ASYMPTOTIC EXPANSION FOR I(FNU,Z) FOR LARGE FNU APPLIED IN
C     -PI/3.LE.ARG(Z).LE.PI/3
C-----------------------------------------------------------------------
      CALL CUNI1(Z, GNU, KODE, 2, CY, NW, NLAST, FNUL, TOL, ELIM, ALIM)
      GO TO 20
   10 CONTINUE
C-----------------------------------------------------------------------
C     ASYMPTOTIC EXPANSION FOR J(FNU,Z*EXP(M*HPI)) FOR LARGE FNU
C     APPLIED IN PI/3.LT.ABS(ARG(Z)).LE.PI/2 WHERE M=+I OR -I
C     AND HPI=PI/2
C-----------------------------------------------------------------------
      CALL CUNI2(Z, GNU, KODE, 2, CY, NW, NLAST, FNUL, TOL, ELIM, ALIM)
   20 CONTINUE
      IF (NW.LT.0) GO TO 50
      IF (NW.NE.0) GO TO 90
      AY = CABS(CY(1))
C----------------------------------------------------------------------
C     SCALE BACKWARD RECURRENCE, BRY(3) IS DEFINED BUT NEVER USED
C----------------------------------------------------------------------
      BRY(1) = 1.0E+3*R1MACH(1)/TOL
      BRY(2) = 1.0E0/BRY(1)
      BRY(3) = BRY(2)
      IFLAG = 2
      ASCLE = BRY(2)
      AX = 1.0E0
      CSCL = CMPLX(AX,0.0E0)
      IF (AY.GT.BRY(1)) GO TO 21
      IFLAG = 1
      ASCLE = BRY(1)
      AX = 1.0E0/TOL
      CSCL = CMPLX(AX,0.0E0)
      GO TO 25
   21 CONTINUE
      IF (AY.LT.BRY(2)) GO TO 25
      IFLAG = 3
      ASCLE = BRY(3)
      AX = TOL
      CSCL = CMPLX(AX,0.0E0)
   25 CONTINUE
      AY = 1.0E0/AX
      CSCR = CMPLX(AY,0.0E0)
```

```
  S1 = CY(2)*CSCL
  S2 = CY(1)*CSCL
  RZ = CMPLX(2.0E0,0.0E0)/Z
  DO 30 I=1,NUI
   ST = S2
   S2 = CMPLX(DFNU+FNUI,0.0E0)*RZ*S2 + S1
   S1 = ST
   FNUI = FNUI - 1.0E0
   IF (IFLAG.GE.3) GO TO 30
   ST = S2*CSCR
   STR = REAL(ST)
   STI = AIMAG(ST)
   STR = ABS(STR)
   STI = ABS(STI)
   STM = AMAX1(STR,STI)
   IF (STM.LE.ASCLE) GO TO 30
   IFLAG = IFLAG+1
   ASCLE = BRY(IFLAG)
   S1 = S1*CSCR
   S2 = ST
   AX = AX*TOL
   AY = 1.0E0/AX
   CSCL = CMPLX(AX,0.0E0)
   CSCR = CMPLX(AY,0.0E0)
   S1 = S1*CSCL
   S2 = S2*CSCL
30 CONTINUE
  Y(N) = S2*CSCR
  IF (N.EQ.1) RETURN
  NL = N - 1
  FNUI = FLOAT(NL)
  K = NL
  DO 40 I=1,NL
   ST = S2
   S2 = CMPLX(FNU+FNUI,0.0E0)*RZ*S2 + S1
   S1 = ST
   ST = S2*CSCR
   Y(K) = ST
   FNUI = FNUI - 1.0E0
   K = K - 1
   IF (IFLAG.GE.3) GO TO 40
   STR = REAL(ST)
   STI = AIMAG(ST)
   STR = ABS(STR)
   STI = ABS(STI)
   STM = AMAX1(STR,STI)
   IF (STM.LE.ASCLE) GO TO 40
   IFLAG = IFLAG+1
   ASCLE = BRY(IFLAG)
   S1 = S1*CSCR
   S2 = ST
   AX = AX*TOL
   AY = 1.0E0/AX
   CSCL = CMPLX(AX,0.0E0)
   CSCR = CMPLX(AY,0.0E0)
   S1 = S1*CSCL
   S2 = S2*CSCL
40 CONTINUE
  RETURN
50 CONTINUE
  NZ = -1
```

```
      IF(NW.EQ.(-2)) NZ=-2
      RETURN
   60 CONTINUE
      IF (IFORM.EQ.2) GO TO 70
C-----------------------------------------------------------------------
C     ASYMPTOTIC EXPANSION FOR I(FNU,Z) FOR LARGE FNU APPLIED IN
C     -PI/3.LE.ARG(Z).LE.PI/3
C-----------------------------------------------------------------------
      CALL CUNI1(Z, FNU, KODE, N, Y, NW, NLAST, FNUL, TOL, ELIM, ALIM)
      GO TO 80
   70 CONTINUE
C-----------------------------------------------------------------------
C     ASYMPTOTIC EXPANSION FOR J(FNU,Z*EXP(M*HPI)) FOR LARGE FNU
C     APPLIED IN PI/3.LT.ABS(ARG(Z)).LE.PI/2 WHERE M=+I OR -I
C     AND HPI=PI/2
C-----------------------------------------------------------------------
      CALL CUNI2(Z, FNU, KODE, N, Y, NW, NLAST, FNUL, TOL, ELIM, ALIM)
   80 CONTINUE
      IF (NW.LT.0) GO TO 50
      NZ = NW
      RETURN
   90 CONTINUE
      NLAST = N
      RETURN
      END
      SUBROUTINE CBUNK(Z, FNU, KODE, MR, N, Y, NZ, TOL, ELIM, ALIM)
C***BEGIN PROLOGUE  CBUNK
C***REFER TO  CBESK,CBESH
C
C     CBUNK COMPUTES THE K BESSEL FUNCTION FOR FNU.GT.FNUL.
C     ACCORDING TO THE UNIFORM ASYMPTOTIC EXPANSION FOR K(FNU,Z)
C     IN CUNK1 AND THE EXPANSION FOR H(2,FNU,Z) IN CUNK2
C
C***ROUTINES CALLED  CUNK1,CUNK2
C***END PROLOGUE  CBUNK
      COMPLEX Y, Z
      REAL ALIM, AX, AY, ELIM, FNU, TOL, XX, YY
      INTEGER KODE, MR, N, NZ
      DIMENSION Y(N)
      NZ = 0
      XX = REAL(Z)
      YY = AIMAG(Z)
      AX = ABS(XX)*1.7321E0
      AY = ABS(YY)
      IF (AY.GT.AX) GO TO 10
C-----------------------------------------------------------------------
C     ASYMPTOTIC EXPANSION FOR K(FNU,Z) FOR LARGE FNU APPLIED IN
C     -PI/3.LE.ARG(Z).LE.PI/3
C-----------------------------------------------------------------------
      CALL CUNK1(Z, FNU, KODE, MR, N, Y, NZ, TOL, ELIM, ALIM)
      GO TO 20
   10 CONTINUE
C-----------------------------------------------------------------------
C     ASYMPTOTIC EXPANSION FOR H(2,FNU,Z*EXP(M*HPI)) FOR LARGE FNU
C     APPLIED IN PI/3.LT.ABS(ARG(Z)).LE.PI/2 WHERE M=+I OR -I
C     AND HPI=PI/2
C-----------------------------------------------------------------------
      CALL CUNK2(Z, FNU, KODE, MR, N, Y, NZ, TOL, ELIM, ALIM)
   20 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE CKSCL(ZR, FNU, N, Y, NZ, RZ, ASCLE, TOL, ELIM)
C***BEGIN PROLOGUE  CKSCL
C***REFER TO  CBKNU,CUNK1,CUNK2
C
C     SET K FUNCTIONS TO ZERO ON UNDERFLOW, CONTINUE RECURRENCE
C     ON SCALED FUNCTIONS UNTIL TWO MEMBERS COME ON SCALE, THEN
C     RETURN WITH MIN(NZ+2,N) VALUES SCALED BY 1/TOL.
C
C***ROUTINES CALLED  CUCHK
C***END PROLOGUE  CKSCL
      COMPLEX CK, CS, CY, CZERO, RZ, S1, S2, Y, ZR, ZD, CELM
      REAL AA, ASCLE, ACS, AS, CSI, CSR, ELIM, FN, FNU, TOL, XX, ZRI,
     * ELM, ALAS, HELIM
      INTEGER I, IC, K, KK, N, NN, NW, NZ
      DIMENSION Y(N), CY(2)
      DATA CZERO / (0.0E0,0.0E0) /
C
      NZ = 0
      IC = 0
      XX = REAL(ZR)
      NN = MIN0(2,N)
      DO 10 I=1,NN
        S1 = Y(I)
        CY(I) = S1
        AS = CABS(S1)
        ACS = -XX + ALOG(AS)
        NZ = NZ + 1
        Y(I) = CZERO
        IF (ACS.LT.(-ELIM)) GO TO 10
        CS = -ZR + CLOG(S1)
        CSR = REAL(CS)
        CSI = AIMAG(CS)
        AA = EXP(CSR)/TOL
        CS = CMPLX(AA,0.0E0)*CMPLX(COS(CSI),SIN(CSI))
        CALL CUCHK(CS, NW, ASCLE, TOL)
        IF (NW.NE.0) GO TO 10
        Y(I) = CS
        NZ = NZ - 1
        IC = I
   10 CONTINUE
      IF (N.EQ.1) RETURN
      IF (IC.GT.1) GO TO 20
      Y(1) = CZERO
      NZ = 2
   20 CONTINUE
      IF (N.EQ.2) RETURN
      IF (NZ.EQ.0) RETURN
      FN = FNU + 1.0E0
      CK = CMPLX(FN,0.0E0)*RZ
      S1 = CY(1)
      S2 = CY(2)
      HELIM = 0.5E0*ELIM
      ELM = EXP(-ELIM)
      CELM = CMPLX(ELM,0.0E0)
      ZRI =AIMAG(ZR)
      ZD = ZR
C
C     FIND TWO CONSECUTIVE Y VALUES ON SCALE. SCALE RECURRENCE IF
C     S2 GETS LARGER THAN EXP(ELIM/2)
C
      DO 30 I=3,N
```

```
         KK = I
         CS = S2
         S2 = CK*S2 + S1
         S1 = CS
         CK = CK + RZ
         AS = CABS(S2)
         ALAS = ALOG(AS)
         ACS = -XX + ALAS
         NZ = NZ + 1
         Y(I) = CZERO
         IF (ACS.LT.(-ELIM)) GO TO 25
         CS = -ZD + CLOG(S2)
         CSR = REAL(CS)
         CSI = AIMAG(CS)
         AA = EXP(CSR)/TOL
         CS = CMPLX(AA,0.0E0)*CMPLX(COS(CSI),SIN(CSI))
         CALL CUCHK(CS, NW, ASCLE, TOL)
         IF (NW.NE.0) GO TO 25
         Y(I) = CS
         NZ = NZ - 1
         IF (IC.EQ.(KK-1)) GO TO 40
         IC = KK
         GO TO 30
 25      CONTINUE
         IF(ALAS.LT.HELIM) GO TO 30
         XX = XX-ELIM
         S1 = S1*CELM
         S2 = S2*CELM
         ZD = CMPLX(XX,ZRI)
 30   CONTINUE
      NZ = N
      IF(IC.EQ.N) NZ=N-1
      GO TO 45
 40   CONTINUE
      NZ = KK - 2
 45   CONTINUE
      DO 50 K=1,NZ
        Y(K) = CZERO
 50   CONTINUE
      RETURN
      END
      SUBROUTINE CMLRI(Z, FNU, KODE, N, Y, NZ, TOL)
C***BEGIN PROLOGUE  CMLRI
C***REFER TO  CBESI,CBESK
C
C     CMLRI COMPUTES THE I BESSEL FUNCTION FOR RE(Z).GE.0.0 BY THE
C     MILLER ALGORITHM NORMALIZED BY A NEUMANN SERIES.
C
C***ROUTINES CALLED  GAMLN,R1MACH
C***END PROLOGUE  CMLRI
      COMPLEX CK, CNORM, CONE, CTWO, CZERO, PT, P1, P2, RZ, SUM, Y, Z
      REAL ACK, AK, AP, AT, AZ, BK, FKAP, FKK, FLAM, FNF, FNU, RHO,
     * RHO2, SCLE, TFNF, TOL, TST, X, GAMLN, R1MACH
      INTEGER I, IAZ, IDUM, IFNU, INU, ITIME, K, KK, KM, KODE, M, N
      DIMENSION Y(N)
      DATA CZERO,CONE,CTWO /(0.0E0,0.0E0),(1.0E0,0.0E0),(2.0E0,0.0E0)/
      SCLE = 1.0E+3*R1MACH(1)/TOL
      NZ=0
      AZ = CABS(Z)
      X = REAL(Z)
      IAZ = INT(AZ)
```

```
      IFNU = INT(FNU)
      INU = IFNU + N - 1
      AT = FLOAT(IAZ) + 1.0E0
      CK = CMPLX(AT,0.0E0)/Z
      RZ = CTWO/Z
      P1 = CZERO
      P2 = CONE
      ACK = (AT+1.0E0)/AZ
      RHO = ACK + SQRT(ACK*ACK-1.0E0)
      RHO2 = RHO*RHO
      TST = (RHO2+RHO2)/((RHO2-1.0E0)*(RHO-1.0E0))
      TST = TST/TOL
C-----------------------------------------------------------------------
C     COMPUTE RELATIVE TRUNCATION ERROR INDEX FOR SERIES
C-----------------------------------------------------------------------
      AK = AT
      DO 10 I=1,80
        PT = P2
        P2 = P1 - CK*P2
        P1 = PT
        CK = CK + RZ
        AP = CABS(P2)
        IF (AP.GT.TST*AK*AK) GO TO 20
        AK = AK + 1.0E0
   10 CONTINUE
      GO TO 110
   20 CONTINUE
      I = I + 1
      K = 0
      IF (INU.LT.IAZ) GO TO 40
C-----------------------------------------------------------------------
C     COMPUTE RELATIVE TRUNCATION ERROR FOR RATIOS
C-----------------------------------------------------------------------
      P1 = CZERO
      P2 = CONE
      AT = FLOAT(INU) + 1.0E0
      CK = CMPLX(AT,0.0E0)/Z
      ACK = AT/AZ
      TST = SQRT(ACK/TOL)
      ITIME = 1
      DO 30 K=1,80
        PT = P2
        P2 = P1 - CK*P2
        P1 = PT
        CK = CK + RZ
        AP = CABS(P2)
        IF (AP.LT.TST) GO TO 30
        IF (ITIME.EQ.2) GO TO 40
        ACK = CABS(CK)
        FLAM = ACK + SQRT(ACK*ACK-1.0E0)
        FKAP = AP/CABS(P1)
        RHO = AMIN1(FLAM,FKAP)
        TST = TST*SQRT(RHO/(RHO*RHO-1.0E0))
        ITIME = 2
   30 CONTINUE
      GO TO 110
   40 CONTINUE
C-----------------------------------------------------------------------
C     BACKWARD RECURRENCE AND SUM NORMALIZING RELATION
C-----------------------------------------------------------------------
      K = K + 1
```

```
      KK = MAX0(I+IAZ,K+INU)
      FKK = FLOAT(KK)
      P1 = CZERO
C-----------------------------------------------------------------------
C     SCALE P2 AND SUM BY SCLE
C-----------------------------------------------------------------------
      P2 = CMPLX(SCLE,0.0E0)
      FNF = FNU - FLOAT(IFNU)
      TFNF = FNF + FNF
      BK = GAMLN(FKK+TFNF+1.0E0,IDUM) - GAMLN(FKK+1.0E0,IDUM)
     *     -GAMLN(TFNF+1.0E0,IDUM)
      BK = EXP(BK)
      SUM = CZERO
      KM = KK - INU
      DO 50 I=1,KM
        PT = P2
        P2 = P1 + CMPLX(FKK+FNF,0.0E0)*RZ*P2
        P1 = PT
        AK = 1.0E0 - TFNF/(FKK+TFNF)
        ACK = BK*AK
        SUM = SUM + CMPLX(ACK+BK,0.0E0)*P1
        BK = ACK
        FKK = FKK - 1.0E0
   50 CONTINUE
      Y(N) = P2
      IF (N.EQ.1) GO TO 70
      DO 60 I=2,N
        PT = P2
        P2 = P1 + CMPLX(FKK+FNF,0.0E0)*RZ*P2
        P1 = PT
        AK = 1.0E0 - TFNF/(FKK+TFNF)
        ACK = BK*AK
        SUM = SUM + CMPLX(ACK+BK,0.0E0)*P1
        BK = ACK
        FKK = FKK - 1.0E0
        M = N - I + 1
        Y(M) = P2
   60 CONTINUE
   70 CONTINUE
      IF (IFNU.LE.0) GO TO 90
      DO 80 I=1,IFNU
        PT = P2
        P2 = P1 + CMPLX(FKK+FNF,0.0E0)*RZ*P2
        P1 = PT
        AK = 1.0E0 - TFNF/(FKK+TFNF)
        ACK = BK*AK
        SUM = SUM + CMPLX(ACK+BK,0.0E0)*P1
        BK = ACK
        FKK = FKK - 1.0E0
   80 CONTINUE
   90 CONTINUE
      PT = Z
      IF (KODE.EQ.2) PT = PT - CMPLX(X,0.0E0)
      P1 = -CMPLX(FNF,0.0E0)*CLOG(RZ) + PT
      AP = GAMLN(1.0E0+FNF,IDUM)
      PT = P1 - CMPLX(AP,0.0E0)
C-----------------------------------------------------------------------
C     THE DIVISION CEXP(PT)/(SUM+P2) IS ALTERED TO AVOID OVERFLOW
C     IN THE DENOMINATOR BY SQUARING LARGE QUANTITIES
C-----------------------------------------------------------------------
      P2 = P2 + SUM
```

```
      AP = CABS(P2)
      P1 = CMPLX(1.0E0/AP,0.0E0)
      CK = CEXP(PT)*P1
      PT = CONJG(P2)*P1
      CNORM = CK*PT
      DO 100 I=1,N
        Y(I) = Y(I)*CNORM
  100 CONTINUE
      RETURN
  110 CONTINUE
      NZ=-2
      RETURN
      END
      SUBROUTINE CRATI(Z, FNU, N, CY, TOL)
C***BEGIN PROLOGUE  CRATI
C***REFER TO  CBESI,CBESK,CBESH
C
C     CRATI COMPUTES RATIOS OF I BESSEL FUNCTIONS BY BACKWARD
C     RECURRENCE.  THE STARTING INDEX IS DETERMINED BY FORWARD
C     RECURRENCE AS DESCRIBED IN J. RES. OF NAT. BUR. OF STANDARDS-B,
C     MATHEMATICAL SCIENCES, VOL 77B, P111-114, SEPTEMBER, 1973,
C     BESSEL FUNCTIONS I AND J OF COMPLEX ARGUMENT AND INTEGER ORDER,
C     BY D. J. SOOKNE.
C
C***ROUTINES CALLED  (NONE)
C***END PROLOGUE  CRATI
      COMPLEX CDFNU, CONE, CY, CZERO, PT, P1, P2, RZ, T1, Z
      REAL AK, AMAGZ, AP1, AP2, ARG, AZ, DFNU, FDNU, FLAM, FNU, FNUP,
     * RAP1, RHO, TEST, TEST1, TOL
      INTEGER I, ID, IDNU, INU, ITIME, K, KK, MAGZ, N
      DIMENSION CY(N)
      DATA CZERO, CONE / (0.0E0,0.0E0), (1.0E0,0.0E0) /
      AZ = CABS(Z)
      INU = INT(FNU)
      IDNU = INU + N - 1
      FDNU = FLOAT(IDNU)
      MAGZ = INT(AZ)
      AMAGZ = FLOAT(MAGZ+1)
      FNUP = AMAX1(AMAGZ,FDNU)
      ID = IDNU - MAGZ - 1
      ITIME = 1
      K = 1
      RZ = (CONE+CONE)/Z
      T1 = CMPLX(FNUP,0.0E0)*RZ
      P2 = -T1
      P1 = CONE
      T1 = T1 + RZ
      IF (ID.GT.0) ID = 0
      AP2 = CABS(P2)
      AP1 = CABS(P1)
C-----------------------------------------------------------------------
C     THE OVERFLOW TEST ON K(FNU+I-1,Z) BEFORE THE CALL TO CBKNX
C     GUARANTEES THAT P2 IS ON SCALE. SCALE TEST1 AND ALL SUBSEQUENT
C     P2 VALUES BY AP1 TO ENSURE THAT AN OVERFLOW DOES NOT OCCUR
C     PREMATURELY.
C-----------------------------------------------------------------------
      ARG = (AP2+AP2)/(AP1*TOL)
      TEST1 = SQRT(ARG)
      TEST = TEST1
      RAP1 = 1.0E0/AP1
      P1 = P1*CMPLX(RAP1,0.0E0)
```

```
      P2 = P2*CMPLX(RAP1,0.0E0)
      AP2 = AP2*RAP1
   10 CONTINUE
      K = K + 1
      AP1 = AP2
      PT = P2
      P2 = P1 - T1*P2
      P1 = PT
      T1 = T1 + RZ
      AP2 = CABS(P2)
      IF (AP1.LE.TEST) GO TO 10
      IF (ITIME.EQ.2) GO TO 20
      AK = CABS(T1)*0.5E0
      FLAM = AK + SQRT(AK*AK-1.0E0)
      RHO = AMIN1(AP2/AP1,FLAM)
      TEST = TEST1*SQRT(RHO/(RHO*RHO-1.0E0))
      ITIME = 2
      GO TO 10
   20 CONTINUE
      KK = K + 1 - ID
      AK = FLOAT(KK)
      DFNU = FNU + FLOAT(N-1)
      CDFNU = CMPLX(DFNU,0.0E0)
      T1 = CMPLX(AK,0.0E0)
      P1 = CMPLX(1.0E0/AP2,0.0E0)
      P2 = CZERO
      DO 30 I=1,KK
        PT = P1
        P1 = RZ*(CDFNU+T1)*P1 + P2
        P2 = PT
        T1 = T1 - CONE
   30 CONTINUE
      IF (REAL(P1).NE.0.0E0 .OR. AIMAG(P1).NE.0.0E0) GO TO 40
      P1 = CMPLX(TOL,TOL)
   40 CONTINUE
      CY(N) = P2/P1
      IF (N.EQ.1) RETURN
      K = N - 1
      AK = FLOAT(K)
      T1 = CMPLX(AK,0.0E0)
      CDFNU = CMPLX(FNU,0.0E0)*RZ
      DO 60 I=2,N
        PT = CDFNU + T1*RZ + CY(K+1)
        IF (REAL(PT).NE.0.0E0 .OR. AIMAG(PT).NE.0.0E0) GO TO 50
        PT = CMPLX(TOL,TOL)
   50   CONTINUE
        CY(K) = CONE/PT
        T1 = T1 - CONE
        K = K - 1
   60 CONTINUE
      RETURN
      END
      SUBROUTINE CS1S2(ZR, S1, S2, NZ, ASCLE, ALIM, IUF)
C***BEGIN PROLOGUE  CS1S2
C***REFER TO  CBESK,CAIRY
C
C     CS1S2 TESTS FOR A POSSIBLE UNDERFLOW RESULTING FROM THE
C     ADDITION OF THE I AND K FUNCTIONS IN THE ANALYTIC CON-
C     TINUATION FORMULA WHERE S1=K FUNCTION AND S2=I FUNCTION.
C     ON KODE=1 THE I AND K FUNCTIONS ARE DIFFERENT ORDERS OF
C     MAGNITUDE, BUT FOR KODE=2 THEY CAN BE OF THE SAME ORDER
```

```
C    OF MAGNITUDE AND THE MAXIMUM MUST BE AT LEAST ONE
C    PRECISION ABOVE THE UNDERFLOW LIMIT.
C
C***ROUTINES CALLED  (NONE)
C***END PROLOGUE  CS1S2
      COMPLEX CZERO, C1, S1, S1D, S2, ZR
      REAL AA, ALIM, ALN, ASCLE, AS1, AS2, XX
      INTEGER IUF, NZ
      DATA CZERO / (0.0E0,0.0E0) /
      NZ = 0
      AS1 = CABS(S1)
      AS2 = CABS(S2)
      AA = REAL(S1)
      ALN = AIMAG(S1)
      IF (AA.EQ.0.0E0 .AND. ALN.EQ.0.0E0) GO TO 10
      IF (AS1.EQ.0.0E0) GO TO 10
      XX = REAL(ZR)
      ALN = -XX - XX + ALOG(AS1)
      S1D = S1
      S1 = CZERO
      AS1 = 0.0E0
      IF (ALN.LT.(-ALIM)) GO TO 10
      C1 = CLOG(S1D) - ZR - ZR
      S1 = CEXP(C1)
      AS1 = CABS(S1)
      IUF = IUF + 1
   10 CONTINUE
      AA = AMAX1(AS1,AS2)
      IF (AA.GT.ASCLE) RETURN
      S1 = CZERO
      S2 = CZERO
      NZ = 1
      IUF = 0
      RETURN
      END
      SUBROUTINE CSERI(Z, FNU, KODE, N, Y, NZ, TOL, ELIM, ALIM)
C***BEGIN PROLOGUE  CSERI
C***REFER TO  CBESI,CBESK
C
C    CSERI COMPUTES THE I BESSEL FUNCTION FOR REAL(Z).GE.0.0 BY
C    MEANS OF THE POWER SERIES FOR LARGE CABS(Z) IN THE
C    REGION CABS(Z).LE.2*SQRT(FNU+1). NZ=0 IS A NORMAL RETURN.
C    NZ.GT.0 MEANS THAT THE LAST NZ COMPONENTS WERE SET TO ZERO
C    DUE TO UNDERFLOW. NZ.LT.0 MEANS UNDERFLOW OCCURRED, BUT THE
C    CONDITION CABS(Z).LE.2*SQRT(FNU+1) WAS VIOLATED AND THE
C    COMPUTATION MUST BE COMPLETED IN ANOTHER ROUTINE WITH N=N-
ABS(NZ).
C
C***ROUTINES CALLED  CUCHK,GAMLN,R1MACH
C***END PROLOGUE  CSERI
      COMPLEX AK1, CK, COEF, CONE, CRSC, CZ, CZERO, HZ, RZ, S1, S2, W,
     * Y, Z
      REAL AA, ACZ, AK, ALIM, ARM, ASCLE, ATOL, AZ, DFNU, ELIM, FNU,
     * FNUP, RAK1, RS, RTR1, S, SS, TOL, X, GAMLN, R1MACH
      INTEGER I, IB, IDUM, IFLAG, IL, K, KODE, L, M, N, NN, NW, NZ
      DIMENSION Y(N), W(2)
      DATA CZERO, CONE / (0.0E0,0.0E0), (1.0E0,0.0E0) /
C
      NZ = 0
      AZ = CABS(Z)
      IF (AZ.EQ.0.0E0) GO TO 150
```

```
    X = REAL(Z)
    ARM = 1.0E+3*R1MACH(1)
    RTR1 = SQRT(ARM)
    CRSC = CMPLX(1.0E0,0.0E0)
    IFLAG = 0
    IF (AZ.LT.ARM) GO TO 140
    HZ = Z*CMPLX(0.5E0,0.0E0)
    CZ = CZERO
    IF (AZ.GT.RTR1) CZ = HZ*HZ
    ACZ = CABS(CZ)
    NN = N
    CK = CLOG(HZ)
  10 CONTINUE
    DFNU = FNU + FLOAT(NN-1)
    FNUP = DFNU + 1.0E0
C-----------------------------------------------------------
C   UNDERFLOW TEST
C-----------------------------------------------------------
    AK1 = CK*CMPLX(DFNU,0.0E0)
    AK = GAMLN(FNUP,IDUM)
    AK1 = AK1 - CMPLX(AK,0.0E0)
    IF (KODE.EQ.2) AK1 = AK1 - CMPLX(X,0.0E0)
    RAK1 = REAL(AK1)
    IF (RAK1.GT.(-ELIM)) GO TO 30
  20 CONTINUE
    NZ = NZ + 1
    Y(NN) = CZERO
    IF (ACZ.GT.DFNU) GO TO 170
    NN = NN - 1
    IF (NN.EQ.0) RETURN
    GO TO 10
  30 CONTINUE
    IF (RAK1.GT.(-ALIM)) GO TO 40
    IFLAG = 1
    SS = 1.0E0/TOL
    CRSC = CMPLX(TOL,0.0E0)
    ASCLE = ARM*SS
  40 CONTINUE
    AK = AIMAG(AK1)
    AA = EXP(RAK1)
    IF (IFLAG.EQ.1) AA = AA*SS
    COEF = CMPLX(AA,0.0E0)*CMPLX(COS(AK),SIN(AK))
    ATOL = TOL*ACZ/FNUP
    IL = MIN0(2,NN)
    DO 80 I=1,IL
      DFNU = FNU + FLOAT(NN-I)
      FNUP = DFNU + 1.0E0
      S1 = CONE
      IF (ACZ.LT.TOL*FNUP) GO TO 60
      AK1 = CONE
      AK = FNUP + 2.0E0
      S = FNUP
      AA = 2.0E0
  50   CONTINUE
      RS = 1.0E0/S
      AK1 = AK1*CZ*CMPLX(RS,0.0E0)
      S1 = S1 + AK1
      S = S + AK
      AK = AK + 2.0E0
      AA = AA*ACZ*RS
      IF (AA.GT.ATOL) GO TO 50
```

```
   60  CONTINUE
         M = NN - I + 1
         S2 = S1*COEF
         W(I) = S2
         IF (IFLAG.EQ.0) GO TO 70
         CALL CUCHK(S2, NW, ASCLE, TOL)
         IF (NW.NE.0) GO TO 20
   70  CONTINUE
         Y(M) = S2*CRSC
         IF (I.NE.IL) COEF = COEF*CMPLX(DFNU,0.0E0)/HZ
   80 CONTINUE
       IF (NN.LE.2) RETURN
       K = NN - 2
       AK = FLOAT(K)
       RZ = (CONE+CONE)/Z
       IF (IFLAG.EQ.1) GO TO 110
       IB = 3
   90 CONTINUE
       DO 100 I=IB,NN
         Y(K) = CMPLX(AK+FNU,0.0E0)*RZ*Y(K+1) + Y(K+2)
         AK = AK - 1.0E0
         K = K - 1
  100 CONTINUE
       RETURN
C------------------------------------------------------------
C    RECUR BACKWARD WITH SCALED VALUES
C------------------------------------------------------------
  110 CONTINUE
C------------------------------------------------------------
C    EXP(-ALIM)=EXP(-ELIM)/TOL=APPROX. ONE PRECISION ABOVE THE
C    UNDERFLOW LIMIT = ASCLE = R1MACH(1)*CSCL*1.0E+3
C------------------------------------------------------------
       S1 = W(1)
       S2 = W(2)
       DO 120 L=3,NN
         CK = S2
         S2 = S1 + CMPLX(AK+FNU,0.0E0)*RZ*S2
         S1 = CK
         CK = S2*CRSC
         Y(K) = CK
         AK = AK - 1.0E0
         K = K - 1
         IF (CABS(CK).GT.ASCLE) GO TO 130
  120 CONTINUE
       RETURN
  130 CONTINUE
       IB = L + 1
       IF (IB.GT.NN) RETURN
       GO TO 90
  140 CONTINUE
       NZ = N
       IF (FNU.EQ.0.0E0) NZ = NZ - 1
  150 CONTINUE
       Y(1) = CZERO
       IF (FNU.EQ.0.0E0) Y(1) = CONE
       IF (N.EQ.1) RETURN
       DO 160 I=2,N
         Y(I) = CZERO
  160 CONTINUE
       RETURN
C------------------------------------------------------------
```

```
C     RETURN WITH NZ.LT.0 IF CABS(Z*Z/4).GT.FNU+N-NZ-1 COMPLETE
C     THE CALCULATION IN CBINU WITH N=N-IABS(NZ)
C-----------------------------------------------------------------------
  170 CONTINUE
      NZ = -NZ
      RETURN
      END
      SUBROUTINE CSHCH(Z, CSH, CCH)
C***BEGIN PROLOGUE  CSHCH
C***REFER TO  CBESK,CBESH
C
C     CSHCH COMPUTES THE COMPLEX HYPERBOLIC FUNCTIONS CSH=SINH(X+I*Y)
C     AND CCH=COSH(X+I*Y), WHERE I**2=-1.
C
C***ROUTINES CALLED  (NONE)
C***END PROLOGUE  CSHCH
      COMPLEX CCH, CSH, Z
      REAL CCHI, CCHR, CH, CN, CSHI, CSHR, SH, SN, X, Y, COSH, SINH
      X = REAL(Z)
      Y = AIMAG(Z)
      SH = SINH(X)
      CH = COSH(X)
      SN = SIN(Y)
      CN = COS(Y)
      CSHR = SH*CN
      CSHI = CH*SN
      CSH = CMPLX(CSHR,CSHI)
      CCHR = CH*CN
      CCHI = SH*SN
      CCH = CMPLX(CCHR,CCHI)
      RETURN
      END
      SUBROUTINE CUCHK(Y, NZ, ASCLE, TOL)
C***BEGIN PROLOGUE  CUCHK
C***REFER TO CSERI,CUOIK,CUNK1,CUNK2,CUNI1,CUNI2,CKSCL
C
C      Y ENTERS AS A SCALED QUANTITY WHOSE MAGNITUDE IS GREATER THAN
C      EXP(-ALIM)=ASCLE=1.0E+3*R1MACH(1)/TOL. THE TEST IS MADE TO SEE
C      IF THE MAGNITUDE OF THE REAL OR IMAGINARY PART WOULD UNDER FLOW
C      WHEN Y IS SCALED (BY TOL) TO ITS PROPER VALUE. Y IS ACCEPTED
C      IF THE UNDERFLOW IS AT LEAST ONE PRECISION BELOW THE MAGNITUDE
C      OF THE LARGEST COMPONENT; OTHERWISE THE PHASE ANGLE DOES NOT
HAVE
C      ABSOLUTE ACCURACY AND AN UNDERFLOW IS ASSUMED.
C
C***ROUTINES CALLED  (NONE)
C***END PROLOGUE  CUCHK
C
      COMPLEX Y
      REAL ASCLE, SS, ST, TOL, YR, YI
      INTEGER NZ
      NZ = 0
      YR = REAL(Y)
      YI = AIMAG(Y)
      YR = ABS(YR)
      YI = ABS(YI)
      ST = AMIN1(YR,YI)
      IF (ST.GT.ASCLE) RETURN
      SS = AMAX1(YR,YI)
      ST=ST/TOL
      IF (SS.LT.ST) NZ = 1
```

```
      RETURN
      END
      SUBROUTINE CUNHJ(Z, FNU, IPMTR, TOL, PHI, ARG, ZETA1, ZETA2,
     * ASUM, BSUM)
C***BEGIN PROLOGUE  CUNHJ
C***REFER TO  CBESI,CBESK
C
C   REFERENCES
C       HANDBOOK OF MATHEMATICAL FUNCTIONS BY M. ABRAMOWITZ AND I.A.
C       STEGUN, AMS55, NATIONAL BUREAU OF STANDARDS, 1965, CHAPTER 9.
C
C       ASYMPTOTICS AND SPECIAL FUNCTIONS BY F.W.J. OLVER, ACADEMIC
C       PRESS, N.Y., 1974, PAGE 420
C
C   ABSTRACT
C       CUNHJ COMPUTES PARAMETERS FOR BESSEL FUNCTIONS C(FNU,Z) =
C       J(FNU,Z), Y(FNU,Z) OR H(I,FNU,Z) I=1,2 FOR LARGE ORDERS FNU
C       BY MEANS OF THE UNIFORM ASYMPTOTIC EXPANSION
C
C       C(FNU,Z)=C1*PHI*( ASUM*AIRY(ARG) + C2*BSUM*DAIRY(ARG) )
C
C       FOR PROPER CHOICES OF C1, C2, AIRY AND DAIRY WHERE AIRY IS
C       AN AIRY FUNCTION AND DAIRY IS ITS DERIVATIVE.
C
C           (2/3)*FNU*ZETA**1.5 = ZETA1-ZETA2,
C
C       ZETA1=0.5*FNU*CLOG((1+W)/(1-W)), ZETA2=FNU*W FOR SCALING
C       PURPOSES IN AIRY FUNCTIONS FROM CAIRY OR CBIRY.
C
C       MCONJ=SIGN OF AIMAG(Z), BUT IS AMBIGUOUS WHEN Z IS REAL AND
C       MUST BE SPECIFIED. IPMTR=0 RETURNS ALL PARAMETERS. IPMTR=
C       1 COMPUTES ALL EXCEPT ASUM AND BSUM.
C
C***ROUTINES CALLED  (NONE)
C***END PROLOGUE  CUNHJ
      COMPLEX ARG, ASUM, BSUM, CFNU, CONE, CR, CZERO, DR, P, PHI,
     * PRZTH, PTFN, RFN13, RTZTA, RZTH, SUMA, SUMB, TFN, T2, UP, W, W2,
     * Z, ZA, ZB, ZC, ZETA, ZETA1, ZETA2, ZTH
      REAL ALFA, ANG, AP, AR, ATOL, AW2, AZTH, BETA, BR, BTOL, C, EX1,
     * EX2, FNU, FN13, FN23, GAMA, HPI, PI, PP, RFNU, RFNU2, THPI, TOL,
     * WI, WR, ZCI, ZCR, ZETAI, ZETAR, ZTHI, ZTHR, ASUMR, ASUMI, BSUMR,
     * BSUMI, TEST, TSTR, TSTI, AC
      INTEGER IAS, IBS, IPMTR, IS, J, JR, JU, K, KMAX, KP1, KS, L, LR,
     * LRP1, L1, L2, M
      DIMENSION AR(14), BR(14), C(105), ALFA(180), BETA(210), GAMA(30),
     * AP(30), P(30), UP(14), CR(14), DR(14)
      DATA AR(1), AR(2), AR(3), AR(4), AR(5), AR(6), AR(7), AR(8),
     1     AR(9), AR(10), AR(11), AR(12), AR(13), AR(14)/
     2     1.00000000000000000E+00,     1.04166666666666667E-01,
     3     8.35503472222222222E-02,     1.28226574556327160E-01,
     4     2.91849026464140464E-01,     8.81627267443757652E-01,
     5     3.32140828186276754E+00,     1.49957629868625547E+01,
     6     7.89230130115865181E+01,     4.74451538868264323E+02,
     7     3.20749009089066193E+03,     2.40865496408740049E+04,
     8     1.98923119169509794E+05,     1.79190200777534383E+06/
      DATA BR(1), BR(2), BR(3), BR(4), BR(5), BR(6), BR(7), BR(8),
     1     BR(9), BR(10), BR(11), BR(12), BR(13), BR(14)/
     2     1.00000000000000000E+00,    -1.45833333333333333E-01,
     3    -9.87413194444444444E-02,    -1.43312053915895062E-01,
     4    -3.17227202678413548E-01,    -9.42429147957120249E-01,
     5    -3.51120304082635426E+00,    -1.57272636203680451E+01,
```

```
6   -8.22814390971859444E+01,   -4.92355370523670524E+02,
7   -3.31621856854797251E+03,   -2.48276742452085896E+04,
8   -2.04526587315129788E+05,   -1.83844491706820990E+06/
 DATA C(1), C(2), C(3), C(4), C(5), C(6), C(7), C(8), C(9), C(10),
1    C(11), C(12), C(13), C(14), C(15), C(16), C(17), C(18),
2    C(19), C(20), C(21), C(22), C(23), C(24)/
3    1.00000000000000000E+00,   -2.08333333333333333E-01,
4    1.25000000000000000E-01,    3.34201388888888889E-01,
5   -4.01041666666666667E-01,    7.03125000000000000E-02,
6   -1.02581259645061728E+00,    1.84646267361111111E+00,
7   -8.91210937500000000E-01,    7.32421875000000000E-02,
8    4.66958442342624743E+00,   -1.12070026162229938E+01,
9    8.78912353515625000E+00,   -2.36408691406250000E+00,
A    1.12152099609375000E-01,   -2.82120725582002449E+01,
B    8.46362176746007346E+01,   -9.18182415432400174E+01,
C    4.25349987453884549E+01,   -7.36879435947963170E+00,
D    2.27108001708984375E-01,    2.12570130039217123E+02,
E   -7.65252468141181642E+02,    1.05999045252799988E+03/
 DATA C(25), C(26), C(27), C(28), C(29), C(30), C(31), C(32),
1    C(33), C(34), C(35), C(36), C(37), C(38), C(39), C(40),
2    C(41), C(42), C(43), C(44), C(45), C(46), C(47), C(48)/
3   -6.99579627376132541E+02,    2.18190511744211590E+02,
4   -2.64914304869515555E+01,    5.72501420974731445E-01,
5   -1.91945766231840700E+03,    8.06172218173730938E+03,
6   -1.35865500064341374E+04,    1.16553933368645332E+04,
7   -5.30564697861340311E+03,    1.20090291321635246E+03,
8   -1.08090919788394656E+02,    1.72772750258445740E+00,
9    2.02042913309661486E+04,   -9.69805983886375135E+04,
A    1.92547001232531532E+05,   -2.03400177280415534E+05,
B    1.22200464983017460E+05,   -4.11926549688975513E+04,
C    7.10951430248936372E+03,   -4.93915304773088012E+02,
D    6.07404200127348304E+00,   -2.42919187900551333E+05,
E    1.31176361466297720E+06,   -2.99801591853810675E+06/
 DATA C(49), C(50), C(51), C(52), C(53), C(54), C(55), C(56),
1    C(57), C(58), C(59), C(60), C(61), C(62), C(63), C(64),
2    C(65), C(66), C(67), C(68), C(69), C(70), C(71), C(72)/
3    3.76327129765640400E+06,   -2.81356322658653411E+06,
4    1.26836527332162478E+06,   -3.31645172484563578E+05,
5    4.52187689813627263E+04,   -2.49983048181120962E+03,
6    2.43805296995560639E+01,    3.28446985307203782E+06,
7   -1.97068191184322269E+07,    5.09526024926646422E+07,
8   -7.41051482115326577E+07,    6.63445122747290267E+07,
9   -3.75671766607633513E+07,    1.32887671664218183E+07,
A   -2.78561812808645469E+06,    3.08186404612662398E+05,
B   -1.38860897537170405E+04,    1.10017140269246738E+02,
C   -4.93292536645099620E+07,    3.25573074185765749E+08,
D   -9.39462359681578403E+08,    1.55359689957058006E+09,
E   -1.62108055210833708E+09,    1.10684281682301447E+09/
 DATA C(73), C(74), C(75), C(76), C(77), C(78), C(79), C(80),
1    C(81), C(82), C(83), C(84), C(85), C(86), C(87), C(88),
2    C(89), C(90), C(91), C(92), C(93), C(94), C(95), C(96)/
3   -4.95889784275030309E+08,    1.42062907797533095E+08,
4   -2.44740627257387285E+07,    2.24376817792244943E+06,
5   -8.40054336030240853E+04,    5.51335896122020586E+02,
6    8.14789096118312115E+08,   -5.86648149205184723E+09,
7    1.86882075092958249E+10,   -3.46320433881587779E+10,
8    4.12801855797539740E+10,   -3.30265997498007231E+10,
9    1.79542137311556001E+10,   -6.56329379261928433E+09,
A    1.55927986487925751E+09,   -2.25105661889415278E+08,
B    1.73951075539781645E+07,   -5.49842327572288687E+05,
C    3.03809051092238427E+03,   -1.46792612476956167E+10,
```

```
D     1.14498237732025810E+11,    -3.99096175224466498E+11,
E     8.19218669548577329E+11,    -1.09837515608122331E+12/
 DATA C(97), C(98), C(99), C(100), C(101), C(102), C(103), C(104),
1     C(105)/
2     1.00815810686538209E+12,    -6.45364869245376503E+11,
3     2.87900649906150589E+11,    -8.78670721780232657E+10,
4     1.76347306068349694E+10,    -2.16716498322379509E+09,
5     1.43157876718888981E+08,    -3.87183344257261262E+06,
6     1.82577554742931747E+04/
 DATA ALFA(1), ALFA(2), ALFA(3), ALFA(4), ALFA(5), ALFA(6),
1     ALFA(7), ALFA(8), ALFA(9), ALFA(10), ALFA(11), ALFA(12),
2     ALFA(13), ALFA(14), ALFA(15), ALFA(16), ALFA(17), ALFA(18),
3     ALFA(19), ALFA(20), ALFA(21), ALFA(22)/
4    -4.44444444444444444E-03,    -9.22077922077922078E-04,
5    -8.84892884892884893E-05,     1.65927687832449737E-04,
6     2.46691372741792910E-04,     2.65995589346254780E-04,
7     2.61824297061500945E-04,     2.48730437344655609E-04,
8     2.32721040083232098E-04,     2.16362485712365082E-04,
9     2.00738858762752355E-04,     1.86267636637545172E-04,
A     1.73060775917876493E-04,     1.61091705929015752E-04,
B     1.50274774160908134E-04,     1.40503497391269794E-04,
C     1.31668816545922806E-04,     1.23667445598253261E-04,
D     1.16405271474737902E-04,     1.09798298372713369E-04,
E     1.03772410422992823E-04,     9.82626078369363448E-05/
 DATA ALFA(23), ALFA(24), ALFA(25), ALFA(26), ALFA(27), ALFA(28),
1     ALFA(29), ALFA(30), ALFA(31), ALFA(32), ALFA(33), ALFA(34),
2     ALFA(35), ALFA(36), ALFA(37), ALFA(38), ALFA(39), ALFA(40),
3     ALFA(41), ALFA(42), ALFA(43), ALFA(44)/
4     9.32120517249503256E-05,     8.85710852478711718E-05,
5     8.42963105715700223E-05,     8.03497548407791151E-05,
6     7.66981345359207388E-05,     7.33122157481777809E-05,
7     7.01662625163141333E-05,     6.72375633790160292E-05,
8     6.93735541354588974E-04,     2.32241745182921654E-04,
9    -1.41986273556691197E-05,    -1.16444931672048640E-04,
A    -1.50803558053048762E-04,    -1.55121924918096223E-04,
B    -1.46809756646465549E-04,    -1.33815503867491367E-04,
C    -1.19744975684254051E-04,    -1.06184319207974020E-04,
D    -9.37699549891194492E-05,    -8.26923045588193274E-05,
E    -7.29374348155221211E-05,    -6.44042357721016283E-05/
 DATA ALFA(45), ALFA(46), ALFA(47), ALFA(48), ALFA(49), ALFA(50),
1     ALFA(51), ALFA(52), ALFA(53), ALFA(54), ALFA(55), ALFA(56),
2     ALFA(57), ALFA(58), ALFA(59), ALFA(60), ALFA(61), ALFA(62),
3     ALFA(63), ALFA(64), ALFA(65), ALFA(66)/
4    -5.69611566009369048E-05,    -5.04731044303561628E-05,
5    -4.48134868008882786E-05,    -3.98688727717598864E-05,
6    -3.55400532972042498E-05,    -3.17414256609022480E-05,
7    -2.83996793904174811E-05,    -2.54522720634870566E-05,
8    -2.28459297164724555E-05,    -2.05352753106480604E-05,
9    -1.84816217627666085E-05,    -1.66519330021393806E-05,
A    -1.50179412980119482E-05,    -1.35554031379040526E-05,
B    -1.22434746473858131E-05,    -1.10641884811308169E-05,
C    -3.54211971457743841E-04,    -1.56161263945159416E-04,
D     3.04465503594936410E-05,     1.30198655773242693E-04,
E     1.67471106699712269E-04,     1.70222587683592569E-04/
 DATA ALFA(67), ALFA(68), ALFA(69), ALFA(70), ALFA(71), ALFA(72),
1     ALFA(73), ALFA(74), ALFA(75), ALFA(76), ALFA(77), ALFA(78),
2     ALFA(79), ALFA(80), ALFA(81), ALFA(82), ALFA(83), ALFA(84),
3     ALFA(85), ALFA(86), ALFA(87), ALFA(88)/
4     1.56501427608594704E-04,     1.36339170977445120E-04,
5     1.14886692029825128E-04,     9.45869093034688111E-05,
6     7.64498419250898258E-05,     6.07570334965197354E-05,
```

```
7     4.74394299290508799E-05,    3.62757512005344297E-05,
8     2.69939714979224901E-05,    1.93210938247939253E-05,
9     1.30056674793963203E-05,    7.82620866744496661E-06,
A     3.59257485819351583E-06,    1.44040049814251817E-07,
B    -2.65396769697939116E-06,   -4.91346867098485910E-06,
C    -6.72739296091248287E-06,   -8.17269379678657923E-06,
D    -9.31304715093561232E-06,   -1.02011418798016441E-05,
E    -1.08805962510592880E-05,   -1.13875481509603555E-05/
 DATA ALFA(89), ALFA(90), ALFA(91), ALFA(92), ALFA(93), ALFA(94),
1     ALFA(95), ALFA(96), ALFA(97), ALFA(98), ALFA(99), ALFA(100),
2     ALFA(101), ALFA(102), ALFA(103), ALFA(104), ALFA(105),
3     ALFA(106), ALFA(107), ALFA(108), ALFA(109), ALFA(110)/
4    -1.17519675674556414E-05,   -1.19987364870944141E-05,
5     3.78194199201772914E-04,    2.02471952761816167E-04,
6    -6.37938506318862408E-05,   -2.38598230603005903E-04,
7    -3.10916256027361568E-04,   -3.13680115247576316E-04,
8    -2.78950273791323387E-04,   -2.28564082619141374E-04,
9    -1.75245280340846749E-04,   -1.25544063060690348E-04,
A    -8.22982872820208365E-05,   -4.62860730588116458E-05,
B    -1.72334302366962267E-05,    5.60690482304602267E-06,
C     2.31395443148286800E-05,    3.62642745856793957E-05,
D     4.58006124490188752E-05,    5.24595294959114050E-05,
E     5.68396208545815266E-05,    5.94349820393104052E-05/
 DATA ALFA(111), ALFA(112), ALFA(113), ALFA(114), ALFA(115),
1     ALFA(116), ALFA(117), ALFA(118), ALFA(119), ALFA(120),
2     ALFA(121), ALFA(122), ALFA(123), ALFA(124), ALFA(125),
3     ALFA(126), ALFA(127), ALFA(128), ALFA(129), ALFA(130)/
4     6.06478527578421742E-05,    6.08023907788436497E-05,
5     6.01577894539460388E-05,    5.89199657344698500E-05,
6     5.72515823777593053E-05,    5.52804375585852577E-05,
7     5.31063773802880170E-05,    5.08069302012325706E-05,
8     4.84418647620094842E-05,    4.60568581607475370E-05,
9    -6.91141397288294174E-04,   -4.29976633058871912E-04,
A     1.83067735980039018E-04,    6.60088147542014144E-04,
B     8.75964969951185931E-04,    8.77335235958235514E-04,
C     7.49369585378990637E-04,    5.63832329756980918E-04,
D     3.68059319971443156E-04,    1.88464535514455599E-04/
 DATA ALFA(131), ALFA(132), ALFA(133), ALFA(134), ALFA(135),
1     ALFA(136), ALFA(137), ALFA(138), ALFA(139), ALFA(140),
2     ALFA(141), ALFA(142), ALFA(143), ALFA(144), ALFA(145),
3     ALFA(146), ALFA(147), ALFA(148), ALFA(149), ALFA(150)/
4     3.70663057664904149E-05,   -8.28520220232137023E-05,
5    -1.72751952869172998E-04,   -2.36314873605872983E-04,
6    -2.77966150694906658E-04,   -3.02079514155456919E-04,
7    -3.12594712643820127E-04,   -3.12872558758067163E-04,
8    -3.05678038466324377E-04,   -2.93226470614557331E-04,
9    -2.77255655582934777E-04,   -2.59103928467031709E-04,
A    -2.39784014396480342E-04,   -2.20048260045422848E-04,
B    -2.00443911094971498E-04,   -1.81358692210970687E-04,
C    -1.63057674478657464E-04,   -1.45712672175205844E-04,
D    -1.29425421983924587E-04,   -1.14245691942445952E-04/
 DATA ALFA(151), ALFA(152), ALFA(153), ALFA(154), ALFA(155),
1     ALFA(156), ALFA(157), ALFA(158), ALFA(159), ALFA(160),
2     ALFA(161), ALFA(162), ALFA(163), ALFA(164), ALFA(165),
3     ALFA(166), ALFA(167), ALFA(168), ALFA(169), ALFA(170)/
4     1.92821964248775885E-03,    1.35592576302022234E-03,
5    -7.17858090421302995E-04,   -2.58084802575270346E-03,
6    -3.49271130826168475E-03,   -3.46986299340960628E-03,
7    -2.82285233351310182E-03,   -1.88103076404891354E-03,
8    -8.89531718383947600E-04,    3.87912102631035228E-06,
9     7.28688540119691412E-04,    1.26566373053457758E-03,
```

```
     A    1.62518158372674427E-03,    1.83203153216373172E-03,
     B    1.91588388990527909E-03,    1.90588846755546138E-03,
     C    1.82798982421825727E-03,    1.70389506421121530E-03,
     D    1.55097127171097686E-03,    1.38261421852276159E-03/
      DATA ALFA(171), ALFA(172), ALFA(173), ALFA(174), ALFA(175),
     1    ALFA(176), ALFA(177), ALFA(178), ALFA(179), ALFA(180)/
     2    1.20881424230064774E-03,    1.03676532638344962E-03,
     3    8.71437918068619115E-04,    7.16080155297701002E-04,
     4    5.72637002558129372E-04,    4.42089819465802277E-04,
     5    3.24724948503090564E-04,    2.20342042730246599E-04,
     6    1.28412898401353882E-04,    4.82005924552095464E-05/
      DATA BETA(1), BETA(2), BETA(3), BETA(4), BETA(5), BETA(6),
     1    BETA(7), BETA(8), BETA(9), BETA(10), BETA(11), BETA(12),
     2    BETA(13), BETA(14), BETA(15), BETA(16), BETA(17), BETA(18),
     3    BETA(19), BETA(20), BETA(21), BETA(22)/
     4    1.79988721413553309E-02,    5.59964911064388073E-03,
     5    2.88501402231132779E-03,    1.80096606761053941E-03,
     6    1.24753110589199202E-03,    9.22878876572938311E-04,
     7    7.14430421727287357E-04,    5.71787281789704872E-04,
     8    4.69431007606481533E-04,    3.93232835462916638E-04,
     9    3.34818889318297664E-04,    2.88952148495751517E-04,
     A    2.52211615549573284E-04,    2.22280580798883327E-04,
     B    1.97541838033062524E-04,    1.76836855019718004E-04,
     C    1.59316899661821081E-04,    1.44347930197333986E-04,
     D    1.31448068119965379E-04,    1.20245444949302884E-04,
     E    1.10449144504599392E-04,    1.01828770740567258E-04/
      DATA BETA(23), BETA(24), BETA(25), BETA(26), BETA(27), BETA(28),
     1    BETA(29), BETA(30), BETA(31), BETA(32), BETA(33), BETA(34),
     2    BETA(35), BETA(36), BETA(37), BETA(38), BETA(39), BETA(40),
     3    BETA(41), BETA(42), BETA(43), BETA(44)/
     4    9.41998224204237509E-05,    8.74130545753834437E-05,
     5    8.13466262162801467E-05,    7.59002269646219339E-05,
     6    7.09906300634153481E-05,    6.65482874842468183E-05,
     7    6.25146958969275078E-05,    5.88403394426251749E-05,
     8   -1.49282953213429172E-03,   -8.78204709546389328E-04,
     9   -5.02916549572034614E-04,   -2.94822138512746025E-04,
     A   -1.75463996970782828E-04,   -1.04008550460816434E-04,
     B   -5.96141953046457895E-05,   -3.12038929076098340E-05,
     C   -1.26089735980230047E-05,   -2.42892608575730389E-07,
     D    8.05996165414273571E-06,    1.36507009262147391E-05,
     E    1.73964125472926261E-05,    1.98672978842133780E-05/
      DATA BETA(45), BETA(46), BETA(47), BETA(48), BETA(49), BETA(50),
     1    BETA(51), BETA(52), BETA(53), BETA(54), BETA(55), BETA(56),
     2    BETA(57), BETA(58), BETA(59), BETA(60), BETA(61), BETA(62),
     3    BETA(63), BETA(64), BETA(65), BETA(66)/
     4    2.14463263790822639E-05,    2.23954659232456514E-05,
     5    2.28967783814712629E-05,    2.30785389811177817E-05,
     6    2.30321976080909144E-05,    2.28236073720348722E-05,
     7    2.25005881105292418E-05,    2.20981015361991429E-05,
     8    2.16418427448103905E-05,    2.11507649256220843E-05,
     9    2.06388749782170737E-05,    2.01165241997081666E-05,
     A    1.95913450141179244E-05,    1.90689367910436740E-05,
     B    1.85533719641636667E-05,    1.80475722259674218E-05,
     C    5.52213076721292790E-04,    4.47932581552384646E-04,
     D    2.79520653992020589E-04,    1.52468156198446602E-04,
     E    6.93271105657043598E-05,    1.76258683069991397E-05/
      DATA BETA(67), BETA(68), BETA(69), BETA(70), BETA(71), BETA(72),
     1    BETA(73), BETA(74), BETA(75), BETA(76), BETA(77), BETA(78),
     2    BETA(79), BETA(80), BETA(81), BETA(82), BETA(83), BETA(84),
     3    BETA(85), BETA(86), BETA(87), BETA(88)/
     4   -1.35744996343269136E-05,   -3.17972413350427135E-05,
```

```
5   -4.18861861696693365E-05,    -4.69004889379141029E-05,
6   -4.87665447413787352E-05,    -4.87010031186735069E-05,
7   -4.74755620890086638E-05,    -4.55813058138628452E-05,
8   -4.33309644511266036E-05,    -4.09230193157750364E-05,
9   -3.84822638603221274E-05,    -3.60857167535410501E-05,
A   -3.37793306123367417E-05,    -3.15888560772109621E-05,
B   -2.95269561750807315E-05,    -2.75978914828335759E-05,
C   -2.58006174666883713E-05,    -2.41308356761280200E-05,
D   -2.25823509518346033E-05,    -2.11479656768912971E-05,
E   -1.98200638885294927E-05,    -1.85909870801065077E-05/
 DATA BETA(89), BETA(90), BETA(91), BETA(92), BETA(93), BETA(94),
1    BETA(95), BETA(96), BETA(97), BETA(98), BETA(99), BETA(100),
2    BETA(101), BETA(102), BETA(103), BETA(104), BETA(105),
3    BETA(106), BETA(107), BETA(108), BETA(109), BETA(110)/
4   -1.74532699844210224E-05,    -1.63997823854497997E-05,
5   -4.74617796559959808E-04,    -4.77864567147321487E-04,
6   -3.20390228067037603E-04,    -1.61105016119962282E-04,
7   -4.25778101285435204E-05,     3.44571294294967503E-05,
8    7.97092684075674924E-05,     1.03138236708272200E-04,
9    1.12466775262204158E-04,     1.13103642108481389E-04,
A    1.08651634848774268E-04,     1.01437951597661973E-04,
B    9.29298396593363896E-05,     8.40293133016089978E-05,
C    7.52727991349134062E-05,     6.69632521975730872E-05,
D    5.92564547323194704E-05,     5.22169308826975567E-05,
E    4.58539485165360646E-05,     4.01445513891486808E-05/
 DATA BETA(111), BETA(112), BETA(113), BETA(114), BETA(115),
1    BETA(116), BETA(117), BETA(118), BETA(119), BETA(120),
2    BETA(121), BETA(122), BETA(123), BETA(124), BETA(125),
3    BETA(126), BETA(127), BETA(128), BETA(129), BETA(130)/
4    3.50481730031328081E-05,     3.05157995034346659E-05,
5    2.64956119950516039E-05,     2.29363633690998152E-05,
6    1.97893056664021636E-05,     1.70091984636412623E-05,
7    1.45547428261524004E-05,     1.23886640995878413E-05,
8    1.04775876076583236E-05,     8.79179954978479373E-06,
9    7.36465810572578444E-04,     8.72790805146193976E-04,
A    6.22614862573135066E-04,     2.85998154194304147E-04,
B    3.84737672879366102E-06,    -1.87906003636971558E-04,
C   -2.97603646594554535E-04,    -3.45998126832656348E-04,
D   -3.53382470916037712E-04,    -3.35715635775048757E-04/
 DATA BETA(131), BETA(132), BETA(133), BETA(134), BETA(135),
1    BETA(136), BETA(137), BETA(138), BETA(139), BETA(140),
2    BETA(141), BETA(142), BETA(143), BETA(144), BETA(145),
3    BETA(146), BETA(147), BETA(148), BETA(149), BETA(150)/
4   -3.04321124789039809E-04,    -2.66722723047612821E-04,
5   -2.27654214122819527E-04,    -1.89922611854562356E-04,
6   -1.55058918599093870E-04,    -1.23778240761873630E-04,
7   -9.62926147717644187E-05,    -7.25178327714425337E-05,
8   -5.22070028895633801E-05,    -3.50347750511900522E-05,
9   -2.06489761035551757E-05,    -8.70106096849767054E-06,
A    1.13698686675100290E-06,     9.16426474122778849E-06,
B    1.56477785428872620E-05,     2.08223629482466847E-05,
C    2.48923381004595156E-05,     2.80340509574146325E-05,
D    3.03987774629861915E-05,     3.21156731406700616E-05/
 DATA BETA(151), BETA(152), BETA(153), BETA(154), BETA(155),
1    BETA(156), BETA(157), BETA(158), BETA(159), BETA(160),
2    BETA(161), BETA(162), BETA(163), BETA(164), BETA(165),
3    BETA(166), BETA(167), BETA(168), BETA(169), BETA(170)/
4   -1.80182191963885708E-03,    -2.43402962938042533E-03,
5   -1.83422663549856802E-03,    -7.62204596354009765E-04,
6    2.39079475256927218E-04,     9.49266117176881141E-04,
7    1.34467449701540359E-03,     1.48457495259449178E-03,
```

```
     8   1.44732339830617591E-03,    1.30268261285657186E-03,
     9   1.10351597375642682E-03,    8.86047440419791759E-04,
     A   6.73073208165665473E-04,    4.77603872856582378E-04,
     B   3.05991926358789362E-04,    1.60315694594721630E-04,
     C   4.00749555270613286E-05,   -5.66607461635251611E-05,
     D  -1.32506186772982638E-04,   -1.90296187989614057E-04/
      DATA BETA(171), BETA(172), BETA(173), BETA(174), BETA(175),
     1     BETA(176), BETA(177), BETA(178), BETA(179), BETA(180),
     2     BETA(181), BETA(182), BETA(183), BETA(184), BETA(185),
     3     BETA(186), BETA(187), BETA(188), BETA(189), BETA(190)/
     4  -2.32811450376937408E-04,   -2.62628811464668841E-04,
     5  -2.82050469867598672E-04,   -2.93081563192861167E-04,
     6  -2.97435962176316616E-04,   -2.96557334239348078E-04,
     7  -2.91647363312090861E-04,   -2.83696203837734166E-04,
     8  -2.73512317095673346E-04,   -2.61750155806768580E-04,
     9   6.38585891212050914E-03,    9.62374215806377941E-03,
     A   7.61878061207001043E-03,    2.83219055545628054E-03,
     B  -2.09841352012720090E-03,   -5.73826764216626498E-03,
     C  -7.70804244495414620E-03,   -8.21011692264844401E-03,
     D  -7.65824520346905413E-03,   -6.47209729391045177E-03/
      DATA BETA(191), BETA(192), BETA(193), BETA(194), BETA(195),
     1     BETA(196), BETA(197), BETA(198), BETA(199), BETA(200),
     2     BETA(201), BETA(202), BETA(203), BETA(204), BETA(205),
     3     BETA(206), BETA(207), BETA(208), BETA(209), BETA(210)/
     4  -4.99132412004966473E-03,   -3.45612289713133280E-03,
     5  -2.01785580014170775E-03,   -7.59430686781961401E-04,
     6   2.84173631523859138E-04,    1.10891667586337403E-03,
     7   1.72901493872728771E-03,    2.16812590802684701E-03,
     8   2.45357710494539735E-03,    2.61281821058334862E-03,
     9   2.67141039656276912E-03,    2.65203073395980430E-03,
     A   2.57411652877287315E-03,    2.45389126236094427E-03,
     B   2.30460058071795494E-03,    2.13684837686712662E-03,
     C   1.95896528478870911E-03,    1.77737008679454412E-03,
     D   1.59690280765839059E-03,    1.42111975664438546E-03/
      DATA GAMA(1), GAMA(2), GAMA(3), GAMA(4), GAMA(5), GAMA(6),
     1     GAMA(7), GAMA(8), GAMA(9), GAMA(10), GAMA(11), GAMA(12),
     2     GAMA(13), GAMA(14), GAMA(15), GAMA(16), GAMA(17), GAMA(18),
     3     GAMA(19), GAMA(20), GAMA(21), GAMA(22)/
     4   6.29960524947436582E-01,    2.51984209978974633E-01,
     5   1.54790300415655846E-01,    1.10713062416159013E-01,
     6   8.57309395527394825E-02,    6.97161316958684292E-02,
     7   5.86085671893713576E-02,    5.04698873536310685E-02,
     8   4.42600580689154809E-02,    3.93720661543509966E-02,
     9   3.54283195924455368E-02,    3.21818857502098231E-02,
     A   2.94646240791157679E-02,    2.71581677112934479E-02,
     B   2.51768272973861779E-02,    2.34570755306078891E-02,
     C   2.19508390134907203E-02,    2.06210828235646240E-02,
     D   1.94388240897880846E-02,    1.83810633800683158E-02,
     E   1.74293213231963172E-02,    1.65685837786612353E-02/
      DATA GAMA(23), GAMA(24), GAMA(25), GAMA(26), GAMA(27), GAMA(28),
     1     GAMA(29), GAMA(30)/
     2   1.57865285987918445E-02,    1.50729501494095594E-02,
     3   1.44193250839954639E-02,    1.38184805735341786E-02,
     4   1.32643378994276568E-02,    1.27517121970498651E-02,
     5   1.22761545318762767E-02,    1.18338262398482403E-02/
      DATA EX1, EX2, HPI, PI, THPI /
     1   3.33333333333333333E-01,    6.66666666666666667E-01,
     2   1.57079632679489662E+00,    3.14159265358979324E+00,
     3   4.71238898038468986E+00/
      DATA CZERO, CONE / (0.0E0,0.0E0), (1.0E0,0.0E0) /
C
```

```
      RFNU = 1.0E0/FNU
C     ZB = Z*CMPLX(RFNU,0.0E0)
C-----------------------------------------------------------------------
C     OVERFLOW TEST (Z/FNU TOO SMALL)
C-----------------------------------------------------------------------
      TSTR = REAL(Z)
      TSTI = AIMAG(Z)
      TEST = R1MACH(1)*1.0E+3
      AC = FNU*TEST
      IF (ABS(TSTR).GT.AC .OR. ABS(TSTI).GT.AC) GO TO 15
      AC = 2.0E0*ABS(ALOG(TEST))+FNU
      ZETA1 = CMPLX(AC,0.0E0)
      ZETA2 = CMPLX(FNU,0.0E0)
      PHI=CONE
      ARG=CONE
      RETURN
   15 CONTINUE
      ZB = Z*CMPLX(RFNU,0.0E0)
      RFNU2 = RFNU*RFNU
C-----------------------------------------------------------------------
C     COMPUTE IN THE FOURTH QUADRANT
C-----------------------------------------------------------------------
      FN13 = FNU**EX1
      FN23 = FN13*FN13
      RFN13 = CMPLX(1.0E0/FN13,0.0E0)
      W2 = CONE - ZB*ZB
      AW2 = CABS(W2)
      IF (AW2.GT.0.25E0) GO TO 130
C-----------------------------------------------------------------------
C     POWER SERIES FOR CABS(W2).LE.0.25E0
C-----------------------------------------------------------------------
      K = 1
      P(1) = CONE
      SUMA = CMPLX(GAMA(1),0.0E0)
      AP(1) = 1.0E0
      IF (AW2.LT.TOL) GO TO 20
      DO 10 K=2,30
        P(K) = P(K-1)*W2
        SUMA = SUMA + P(K)*CMPLX(GAMA(K),0.0E0)
        AP(K) = AP(K-1)*AW2
        IF (AP(K).LT.TOL) GO TO 20
   10 CONTINUE
      K = 30
   20 CONTINUE
      KMAX = K
      ZETA = W2*SUMA
      ARG = ZETA*CMPLX(FN23,0.0E0)
      ZA = CSQRT(SUMA)
      ZETA2 = CSQRT(W2)*CMPLX(FNU,0.0E0)
      ZETA1 = ZETA2*(CONE+ZETA*ZA*CMPLX(EX2,0.0E0))
      ZA = ZA + ZA
      PHI = CSQRT(ZA)*RFN13
      IF (IPMTR.EQ.1) GO TO 120
C-----------------------------------------------------------------------
C     SUM SERIES FOR ASUM AND BSUM
C-----------------------------------------------------------------------
      SUMB = CZERO
      DO 30 K=1,KMAX
        SUMB = SUMB + P(K)*CMPLX(BETA(K),0.0E0)
   30 CONTINUE
      ASUM = CZERO
```

```
      BSUM = SUMB
      L1 = 0
      L2 = 30
      BTOL = TOL*CABS(BSUM)
      ATOL = TOL
      PP = 1.0E0
      IAS = 0
      IBS = 0
      IF (RFNU2.LT.TOL) GO TO 110
      DO 100 IS=2,7
       ATOL = ATOL/RFNU2
       PP = PP*RFNU2
       IF (IAS.EQ.1) GO TO 60
       SUMA = CZERO
       DO 40 K=1,KMAX
        M = L1 + K
        SUMA = SUMA + P(K)*CMPLX(ALFA(M),0.0E0)
        IF (AP(K).LT.ATOL) GO TO 50
   40   CONTINUE
   50   CONTINUE
       ASUM = ASUM + SUMA*CMPLX(PP,0.0E0)
       IF (PP.LT.TOL) IAS = 1
   60   CONTINUE
       IF (IBS.EQ.1) GO TO 90
       SUMB = CZERO
       DO 70 K=1,KMAX
        M = L2 + K
        SUMB = SUMB + P(K)*CMPLX(BETA(M),0.0E0)
        IF (AP(K).LT.ATOL) GO TO 80
   70   CONTINUE
   80   CONTINUE
       BSUM = BSUM + SUMB*CMPLX(PP,0.0E0)
       IF (PP.LT.BTOL) IBS = 1
   90   CONTINUE
       IF (IAS.EQ.1 .AND. IBS.EQ.1) GO TO 110
       L1 = L1 + 30
       L2 = L2 + 30
  100 CONTINUE
  110 CONTINUE
      ASUM = ASUM + CONE
      PP = RFNU*REAL(RFN13)
      BSUM = BSUM*CMPLX(PP,0.0E0)
  120 CONTINUE
      RETURN
C-----------------------------------------------------------------------
C     CABS(W2).GT.0.25E0
C-----------------------------------------------------------------------
  130 CONTINUE
      W = CSQRT(W2)
      WR = REAL(W)
      WI = AIMAG(W)
      IF (WR.LT.0.0E0) WR = 0.0E0
      IF (WI.LT.0.0E0) WI = 0.0E0
      W = CMPLX(WR,WI)
      ZA = (CONE+W)/ZB
      ZC = CLOG(ZA)
      ZCR = REAL(ZC)
      ZCI = AIMAG(ZC)
      IF (ZCI.LT.0.0E0) ZCI = 0.0E0
      IF (ZCI.GT.HPI) ZCI = HPI
      IF (ZCR.LT.0.0E0) ZCR = 0.0E0
```

```
      ZC = CMPLX(ZCR,ZCI)
      ZTH = (ZC-W)*CMPLX(1.5E0,0.0E0)
      CFNU = CMPLX(FNU,0.0E0)
      ZETA1 = ZC*CFNU
      ZETA2 = W*CFNU
      AZTH = CABS(ZTH)
      ZTHR = REAL(ZTH)
      ZTHI = AIMAG(ZTH)
      ANG = THPI
      IF (ZTHR.GE.0.0E0 .AND. ZTHI.LT.0.0E0) GO TO 140
      ANG = HPI
      IF (ZTHR.EQ.0.0E0) GO TO 140
      ANG = ATAN(ZTHI/ZTHR)
      IF (ZTHR.LT.0.0E0) ANG = ANG + PI
  140 CONTINUE
      PP = AZTH**EX2
      ANG = ANG*EX2
      ZETAR = PP*COS(ANG)
      ZETAI = PP*SIN(ANG)
      IF (ZETAI.LT.0.0E0) ZETAI = 0.0E0
      ZETA = CMPLX(ZETAR,ZETAI)
      ARG = ZETA*CMPLX(FN23,0.0E0)
      RTZTA = ZTH/ZETA
      ZA = RTZTA/W
      PHI = CSQRT(ZA+ZA)*RFN13
      IF (IPMTR.EQ.1) GO TO 120
      TFN = CMPLX(RFNU,0.0E0)/W
      RZTH = CMPLX(RFNU,0.0E0)/ZTH
      ZC = RZTH*CMPLX(AR(2),0.0E0)
      T2 = CONE/W2
      UP(2) = (T2*CMPLX(C(2),0.0E0)+CMPLX(C(3),0.0E0))*TFN
      BSUM = UP(2) + ZC
      ASUM = CZERO
      IF (RFNU.LT.TOL) GO TO 220
      PRZTH = RZTH
      PTFN = TFN
      UP(1) = CONE
      PP = 1.0E0
      BSUMR = REAL(BSUM)
      BSUMI = AIMAG(BSUM)
      BTOL = TOL*(ABS(BSUMR)+ABS(BSUMI))
      KS = 0
      KP1 = 2
      L = 3
      IAS = 0
      IBS = 0
      DO 210 LR=2,12,2
        LRP1 = LR + 1
C-----------------------------------------------------------------------
C     COMPUTE TWO ADDITIONAL CR, DR, AND UP FOR TWO MORE TERMS IN
C     NEXT SUMA AND SUMB
C-----------------------------------------------------------------------
        DO 160 K=LR,LRP1
          KS = KS + 1
          KP1 = KP1 + 1
          L = L + 1
          ZA = CMPLX(C(L),0.0E0)
          DO 150 J=2,KP1
            L = L + 1
            ZA = ZA*T2 + CMPLX(C(L),0.0E0)
  150     CONTINUE
```

```
          PTFN = PTFN*TFN
          UP(KP1) = PTFN*ZA
          CR(KS) = PRZTH*CMPLX(BR(KS+1),0.0E0)
          PRZTH = PRZTH*RZTH
          DR(KS) = PRZTH*CMPLX(AR(KS+2),0.0E0)
  160   CONTINUE
        PP = PP*RFNU2
        IF (IAS.EQ.1) GO TO 180
        SUMA = UP(LRP1)
        JU = LRP1
        DO 170 JR=1,LR
          JU = JU - 1
          SUMA = SUMA + CR(JR)*UP(JU)
  170   CONTINUE
        ASUM = ASUM + SUMA
        ASUMR = REAL(ASUM)
        ASUMI = AIMAG(ASUM)
        TEST = ABS(ASUMR) + ABS(ASUMI)
        IF (PP.LT.TOL .AND. TEST.LT.TOL) IAS = 1
  180   CONTINUE
        IF (IBS.EQ.1) GO TO 200
        SUMB = UP(LR+2) + UP(LRP1)*ZC
        JU = LRP1
        DO 190 JR=1,LR
          JU = JU - 1
          SUMB = SUMB + DR(JR)*UP(JU)
  190   CONTINUE
        BSUM = BSUM + SUMB
        BSUMR = REAL(BSUM)
        BSUMI = AIMAG(BSUM)
        TEST = ABS(BSUMR) + ABS(BSUMI)
        IF (PP.LT.BTOL .AND. TEST.LT.TOL) IBS = 1
  200   CONTINUE
        IF (IAS.EQ.1 .AND. IBS.EQ.1) GO TO 220
  210 CONTINUE
  220 CONTINUE
      ASUM = ASUM + CONE
      BSUM = -BSUM*RFN13/RTZTA
      GO TO 120
      END
      SUBROUTINE CUNI1(Z, FNU, KODE, N, Y, NZ, NLAST, FNUL, TOL, ELIM,
     * ALIM)
C***BEGIN PROLOGUE  CUNI1
C***REFER TO  CBESI,CBESK
C
C     CUNI1 COMPUTES I(FNU,Z)  BY MEANS OF THE UNIFORM ASYMPTOTIC
C     EXPANSION FOR I(FNU,Z) IN -PI/3.LE.ARG Z.LE.PI/3.
C
C     FNUL IS THE SMALLEST ORDER PERMITTED FOR THE ASYMPTOTIC
C     EXPANSION. NLAST=0 MEANS ALL OF THE Y VALUES WERE SET.
C     NLAST.NE.0 IS THE NUMBER LEFT TO BE COMPUTED BY ANOTHER
C     FORMULA FOR ORDERS FNU TO FNU+NLAST-1 BECAUSE FNU+NLAST-
1.LT.FNUL.
C     Y(I)=CZERO FOR I=NLAST+1,N
C
C***ROUTINES CALLED  CUCHK,CUNIK,CUOIK,R1MACH
C***END PROLOGUE  CUNI1
      COMPLEX CFN, CONE, CRSC, CSCL, CSR, CSS, CWRK, CZERO, C1, C2,
     * PHI, RZ, SUM, S1, S2, Y, Z, ZETA1, ZETA2, CY
      REAL ALIM, APHI, ASCLE, BRY, C2I, C2M, C2R, ELIM, FN, FNU, FNUL,
     * RS1, TOL, YY, R1MACH
```

```
      INTEGER I, IFLAG, INIT, K, KODE, M, N, ND, NLAST, NN, NUF, NW, NZ
      DIMENSION BRY(3), Y(N), CWRK(16), CSS(3), CSR(3), CY(2)
      DATA CZERO, CONE / (0.0E0,0.0E0), (1.0E0,0.0E0) /
C
      NZ = 0
      ND = N
      NLAST = 0
C-----------------------------------------------------------------------
C     COMPUTED VALUES WITH EXPONENTS BETWEEN ALIM AND ELIM IN MAG-
C     NITUDE ARE SCALED TO KEEP INTERMEDIATE ARITHMETIC ON SCALE,
C     EXP(ALIM)=EXP(ELIM)*TOL
C-----------------------------------------------------------------------
      CSCL = CMPLX(1.0E0/TOL,0.0E0)
      CRSC = CMPLX(TOL,0.0E0)
      CSS(1) = CSCL
      CSS(2) = CONE
      CSS(3) = CRSC
      CSR(1) = CRSC
      CSR(2) = CONE
      CSR(3) = CSCL
      BRY(1) = 1.0E+3*R1MACH(1)/TOL
C-----------------------------------------------------------------------
C     CHECK FOR UNDERFLOW AND OVERFLOW ON FIRST MEMBER
C-----------------------------------------------------------------------
      FN = AMAX1(FNU,1.0E0)
      INIT = 0
      CALL CUNIK(Z, FN, 1, 1, TOL, INIT, PHI, ZETA1, ZETA2, SUM, CWRK)
      IF (KODE.EQ.1) GO TO 10
      CFN = CMPLX(FN,0.0E0)
      S1 = -ZETA1 + CFN*(CFN/(Z+ZETA2))
      GO TO 20
   10 CONTINUE
      S1 = -ZETA1 + ZETA2
   20 CONTINUE
      RS1 = REAL(S1)
      IF (ABS(RS1).GT.ELIM) GO TO 130
   30 CONTINUE
      NN = MIN0(2,ND)
      DO 80 I=1,NN
        FN = FNU + FLOAT(ND-I)
        INIT = 0
        CALL CUNIK(Z, FN, 1, 0, TOL, INIT, PHI, ZETA1, ZETA2, SUM, CWRK)
        IF (KODE.EQ.1) GO TO 40
        CFN = CMPLX(FN,0.0E0)
        YY = AIMAG(Z)
        S1 = -ZETA1 + CFN*(CFN/(Z+ZETA2)) + CMPLX(0.0E0,YY)
        GO TO 50
   40   CONTINUE
        S1 = -ZETA1 + ZETA2
   50   CONTINUE
C-----------------------------------------------------------------------
C     TEST FOR UNDERFLOW AND OVERFLOW
C-----------------------------------------------------------------------
        RS1 = REAL(S1)
        IF (ABS(RS1).GT.ELIM) GO TO 110
        IF (I.EQ.1) IFLAG = 2
        IF (ABS(RS1).LT.ALIM) GO TO 60
C-----------------------------------------------------------------------
C     REFINE  TEST AND SCALE
C-----------------------------------------------------------------------
        APHI = CABS(PHI)
```

```
       RS1 = RS1 + ALOG(APHI)
       IF (ABS(RS1).GT.ELIM) GO TO 110
       IF (I.EQ.1) IFLAG = 1
       IF (RS1.LT.0.0E0) GO TO 60
       IF (I.EQ.1) IFLAG = 3
  60   CONTINUE
C------------------------------------------------------------
C     SCALE S1 IF CABS(S1).LT.ASCLE
C------------------------------------------------------------
       S2 = PHI*SUM
       C2R = REAL(S1)
       C2I = AIMAG(S1)
       C2M = EXP(C2R)*REAL(CSS(IFLAG))
       S1 = CMPLX(C2M,0.0E0)*CMPLX(COS(C2I),SIN(C2I))
       S2 = S2*S1
       IF (IFLAG.NE.1) GO TO 70
       CALL CUCHK(S2, NW, BRY(1), TOL)
       IF (NW.NE.0) GO TO 110
  70   CONTINUE
       M = ND - I + 1
       CY(I) = S2
       Y(M) = S2*CSR(IFLAG)
  80 CONTINUE
     IF (ND.LE.2) GO TO 100
     RZ = CMPLX(2.0E0,0.0E0)/Z
     BRY(2) = 1.0E0/BRY(1)
     BRY(3) = R1MACH(2)
     S1 = CY(1)
     S2 = CY(2)
     C1 = CSR(IFLAG)
     ASCLE = BRY(IFLAG)
     K = ND - 2
     FN = FLOAT(K)
     DO 90 I=3,ND
       C2 = S2
       S2 = S1 + CMPLX(FNU+FN,0.0E0)*RZ*S2
       S1 = C2
       C2 = S2*C1
       Y(K) = C2
       K = K - 1
       FN = FN - 1.0E0
       IF (IFLAG.GE.3) GO TO 90
       C2R = REAL(C2)
       C2I = AIMAG(C2)
       C2R = ABS(C2R)
       C2I = ABS(C2I)
       C2M = AMAX1(C2R,C2I)
       IF (C2M.LE.ASCLE) GO TO 90
       IFLAG = IFLAG + 1
       ASCLE = BRY(IFLAG)
       S1 = S1*C1
       S2 = C2
       S1 = S1*CSS(IFLAG)
       S2 = S2*CSS(IFLAG)
       C1 = CSR(IFLAG)
  90 CONTINUE
 100 CONTINUE
     RETURN
C------------------------------------------------------------
C     SET UNDERFLOW AND UPDATE PARAMETERS
C------------------------------------------------------------
```

```
  110 CONTINUE
      IF (RS1.GT.0.0E0) GO TO 120
      Y(ND) = CZERO
      NZ = NZ + 1
      ND = ND - 1
      IF (ND.EQ.0) GO TO 100
      CALL CUOIK(Z, FNU, KODE, 1, ND, Y, NUF, TOL, ELIM, ALIM)
      IF (NUF.LT.0) GO TO 120
      ND = ND - NUF
      NZ = NZ + NUF
      IF (ND.EQ.0) GO TO 100
      FN = FNU + FLOAT(ND-1)
      IF (FN.GE.FNUL) GO TO 30
      NLAST = ND
      RETURN
  120 CONTINUE
      NZ = -1
      RETURN
  130 CONTINUE
      IF (RS1.GT.0.0E0) GO TO 120
      NZ = N
      DO 140 I=1,N
        Y(I) = CZERO
  140 CONTINUE
      RETURN
      END
      SUBROUTINE CUNI2(Z, FNU, KODE, N, Y, NZ, NLAST, FNUL, TOL, ELIM,
     * ALIM)
C***BEGIN PROLOGUE  CUNI2
C***REFER TO  CBESI,CBESK
C
C     CUNI2 COMPUTES I(FNU,Z) IN THE RIGHT HALF PLANE BY MEANS OF
C     UNIFORM ASYMPTOTIC EXPANSION FOR J(FNU,ZN) WHERE ZN IS Z*I
C     OR -Z*I AND ZN IS IN THE RIGHT HALF PLANE ALSO.
C
C     FNUL IS THE SMALLEST ORDER PERMITTED FOR THE ASYMPTOTIC
C     EXPANSION. NLAST=0 MEANS ALL OF THE Y VALUES WERE SET.
C     NLAST.NE.0 IS THE NUMBER LEFT TO BE COMPUTED BY ANOTHER
C     FORMULA FOR ORDERS FNU TO FNU+NLAST-1 BECAUSE FNU+NLAST-
1.LT.FNUL.
C     Y(I)=CZERO FOR I=NLAST+1,N
C
C***ROUTINES CALLED  CAIRY,CUCHK,CUNHJ,CUOIK,R1MACH
C***END PROLOGUE  CUNI2
      COMPLEX AI, ARG, ASUM, BSUM, CFN, CI, CID, CIP, CONE, CRSC, CSCL,
     * CSR, CSS, CY, CZERO, C1, C2, DAI, PHI, RZ, S1, S2, Y, Z, ZB,
     * ZETA1, ZETA2, ZN, ZAR
      REAL AARG, AIC, ALIM, ANG, APHI, ASCLE, AY, BRY, CAR, C2I, C2M,
     * C2R, ELIM, FN, FNU, FNUL, HPI, RS1, SAR, TOL, YY, R1MACH
      INTEGER I, IFLAG, IN, INU, J, K, KODE, N, NAI, ND, NDAI, NLAST,
     * NN, NUF, NW, NZ, IDUM
      DIMENSION BRY(3), Y(N), CIP(4), CSS(3), CSR(3), CY(2)
      DATA CZERO,CONE,CI/(0.0E0,0.0E0),(1.0E0,0.0E0),(0.0E0,1.0E0)/
      DATA CIP(1),CIP(2),CIP(3),CIP(4)/
     1 (1.0E0,0.0E0), (0.0E0,1.0E0), (-1.0E0,0.0E0), (0.0E0,-1.0E0)/
      DATA HPI, AIC  /
     1      1.57079632679489662E+00,     1.265512123484645396E+00/
C
      NZ = 0
      ND = N
      NLAST = 0
```

```
C-----------------------------------------------------------------------
C     COMPUTED VALUES WITH EXPONENTS BETWEEN ALIM AND ELIM IN MAG-
C     NITUDE ARE SCALED TO KEEP INTERMEDIATE ARITHMETIC ON SCALE,
C     EXP(ALIM)=EXP(ELIM)*TOL
C-----------------------------------------------------------------------
      CSCL = CMPLX(1.0E0/TOL,0.0E0)
      CRSC = CMPLX(TOL,0.0E0)
      CSS(1) = CSCL
      CSS(2) = CONE
      CSS(3) = CRSC
      CSR(1) = CRSC
      CSR(2) = CONE
      CSR(3) = CSCL
      BRY(1) = 1.0E+3*R1MACH(1)/TOL
      YY = AIMAG(Z)
C-----------------------------------------------------------------------
C     ZN IS IN THE RIGHT HALF PLANE AFTER ROTATION BY CI OR -CI
C-----------------------------------------------------------------------
      ZN = -Z*CI
      ZB = Z
      CID = -CI
      INU = INT(FNU)
      ANG = HPI*(FNU-FLOAT(INU))
      CAR = COS(ANG)
      SAR = SIN(ANG)
      C2 = CMPLX(CAR,SAR)
      ZAR = C2
      IN = INU + N - 1
      IN = MOD(IN,4)
      C2 = C2*CIP(IN+1)
      IF (YY.GT.0.0E0) GO TO 10
      ZN = CONJG(-ZN)
      ZB = CONJG(ZB)
      CID = -CID
      C2 = CONJG(C2)
   10 CONTINUE
C-----------------------------------------------------------------------
C     CHECK FOR UNDERFLOW AND OVERFLOW ON FIRST MEMBER
C-----------------------------------------------------------------------
      FN = AMAX1(FNU,1.0E0)
      CALL CUNHJ(ZN, FN, 1, TOL, PHI, ARG, ZETA1, ZETA2, ASUM, BSUM)
      IF (KODE.EQ.1) GO TO 20
      CFN = CMPLX(FNU,0.0E0)
      S1 = -ZETA1 + CFN*(CFN/(ZB+ZETA2))
      GO TO 30
   20 CONTINUE
      S1 = -ZETA1 + ZETA2
   30 CONTINUE
      RS1 = REAL(S1)
      IF (ABS(RS1).GT.ELIM) GO TO 150
   40 CONTINUE
      NN = MIN0(2,ND)
      DO 90 I=1,NN
        FN = FNU + FLOAT(ND-I)
        CALL CUNHJ(ZN, FN, 0, TOL, PHI, ARG, ZETA1, ZETA2, ASUM, BSUM)
        IF (KODE.EQ.1) GO TO 50
        CFN = CMPLX(FN,0.0E0)
        AY = ABS(YY)
        S1 = -ZETA1 + CFN*(CFN/(ZB+ZETA2)) + CMPLX(0.0E0,AY)
        GO TO 60
   50   CONTINUE
```

```
      S1 = -ZETA1 + ZETA2
  60  CONTINUE
C-----------------------------------------------------------
C     TEST FOR UNDERFLOW AND OVERFLOW
C-----------------------------------------------------------
      RS1 = REAL(S1)
      IF (ABS(RS1).GT.ELIM) GO TO 120
      IF (I.EQ.1) IFLAG = 2
      IF (ABS(RS1).LT.ALIM) GO TO 70
C-----------------------------------------------------------
C     REFINE  TEST AND SCALE
C-----------------------------------------------------------
C-----------------------------------------------------------
      APHI = CABS(PHI)
      AARG = CABS(ARG)
      RS1 = RS1 + ALOG(APHI) - 0.25E0*ALOG(AARG) - AIC
      IF (ABS(RS1).GT.ELIM) GO TO 120
      IF (I.EQ.1) IFLAG = 1
      IF (RS1.LT.0.0E0) GO TO 70
      IF (I.EQ.1) IFLAG = 3
  70  CONTINUE
C-----------------------------------------------------------
C     SCALE S1 TO KEEP INTERMEDIATE ARITHMETIC ON SCALE NEAR
C     EXPONENT EXTREMES
C-----------------------------------------------------------
      CALL CAIRY(ARG, 0, 2, AI, NAI, IDUM)
      CALL CAIRY(ARG, 1, 2, DAI, NDAI, IDUM)
      S2 = PHI*(AI*ASUM+DAI*BSUM)
      C2R = REAL(S1)
      C2I = AIMAG(S1)
      C2M = EXP(C2R)*REAL(CSS(IFLAG))
      S1 = CMPLX(C2M,0.0E0)*CMPLX(COS(C2I),SIN(C2I))
      S2 = S2*S1
      IF (IFLAG.NE.1) GO TO 80
      CALL CUCHK(S2, NW, BRY(1), TOL)
      IF (NW.NE.0) GO TO 120
  80  CONTINUE
      IF (YY.LE.0.0E0) S2 = CONJG(S2)
      J = ND - I + 1
      S2 = S2*C2
      CY(I) = S2
      Y(J) = S2*CSR(IFLAG)
      C2 = C2*CID
  90 CONTINUE
     IF (ND.LE.2) GO TO 110
     RZ = CMPLX(2.0E0,0.0E0)/Z
     BRY(2) = 1.0E0/BRY(1)
     BRY(3) = R1MACH(2)
     S1 = CY(1)
     S2 = CY(2)
     C1 = CSR(IFLAG)
     ASCLE = BRY(IFLAG)
     K = ND - 2
     FN = FLOAT(K)
     DO 100 I=3,ND
      C2 = S2
      S2 = S1 + CMPLX(FNU+FN,0.0E0)*RZ*S2
      S1 = C2
      C2 = S2*C1
      Y(K) = C2
      K = K - 1
```

```
    FN = FN - 1.0E0
    IF (IFLAG.GE.3) GO TO 100
    C2R = REAL(C2)
    C2I = AIMAG(C2)
    C2R = ABS(C2R)
    C2I = ABS(C2I)
    C2M = AMAX1(C2R,C2I)
    IF (C2M.LE.ASCLE) GO TO 100
    IFLAG = IFLAG + 1
    ASCLE = BRY(IFLAG)
    S1 = S1*C1
    S2 = C2
    S1 = S1*CSS(IFLAG)
    S2 = S2*CSS(IFLAG)
    C1 = CSR(IFLAG)
 100 CONTINUE
 110 CONTINUE
   RETURN
 120 CONTINUE
   IF (RS1.GT.0.0E0) GO TO 140
C-----------------------------------------------------------------------
C    SET UNDERFLOW AND UPDATE PARAMETERS
C-----------------------------------------------------------------------
   Y(ND) = CZERO
   NZ = NZ + 1
   ND = ND - 1
   IF (ND.EQ.0) GO TO 110
   CALL CUOIK(Z, FNU, KODE, 1, ND, Y, NUF, TOL, ELIM, ALIM)
   IF (NUF.LT.0) GO TO 140
   ND = ND - NUF
   NZ = NZ + NUF
   IF (ND.EQ.0) GO TO 110
   FN = FNU + FLOAT(ND-1)
   IF (FN.LT.FNUL) GO TO 130
C     FN = AIMAG(CID)
C     J = NUF + 1
C     K = MOD(J,4) + 1
C     S1 = CIP(K)
C     IF (FN.LT.0.0E0) S1 = CONJG(S1)
C     C2 = C2*S1
   IN = INU + ND - 1
   IN = MOD(IN,4) + 1
   C2 = ZAR*CIP(IN)
   IF (YY.LE.0.0E0)C2=CONJG(C2)
   GO TO 40
 130 CONTINUE
   NLAST = ND
   RETURN
 140 CONTINUE
   NZ = -1
   RETURN
 150 CONTINUE
   IF (RS1.GT.0.0E0) GO TO 140
   NZ = N
   DO 160 I=1,N
    Y(I) = CZERO
 160 CONTINUE
   RETURN
   END
   SUBROUTINE CUNIK(ZR, FNU, IKFLG, IPMTR, TOL, INIT, PHI, ZETA1,
  * ZETA2, SUM, CWRK)
```

```
C***BEGIN PROLOGUE  CUNIK
C***REFER TO  CBESI,CBESK
C
C      CUNIK COMPUTES PARAMETERS FOR THE UNIFORM ASYMPTOTIC
C      EXPANSIONS OF THE I AND K FUNCTIONS ON IKFLG= 1 OR 2
C      RESPECTIVELY BY
C
C      W(FNU,ZR) = PHI*EXP(ZETA)*SUM
C
C      WHERE      ZETA=-ZETA1 + ZETA2      OR
C                  ZETA1 - ZETA2
C
C      THE FIRST CALL MUST HAVE INIT=0. SUBSEQUENT CALLS WITH THE
C      SAME ZR AND FNU WILL RETURN THE I OR K FUNCTION ON IKFLG=
C      1 OR 2 WITH NO CHANGE IN INIT. CWRK IS A COMPLEX WORK
C      ARRAY. IPMTR=0 COMPUTES ALL PARAMETERS. IPMTR=1 COMPUTES PHI,
C      ZETA1,ZETA2.
C
C***ROUTINES CALLED  (NONE)
C***END PROLOGUE  CUNIK
      COMPLEX CFN, CON, CONE, CRFN, CWRK, CZERO, PHI, S, SR, SUM, T,
     * T2, ZETA1, ZETA2, ZN, ZR
      REAL AC, C, FNU, RFN, TEST, TOL, TSTR, TSTI
      INTEGER I, IKFLG, INIT, IPMTR, J, K, L
      DIMENSION C(120), CWRK(16), CON(2)
      DATA CZERO, CONE / (0.0E0,0.0E0), (1.0E0,0.0E0) /
      DATA CON(1), CON(2)  /
     1(3.98942280401432678E-01,0.0E0),(1.25331413731550025E+00,0.0E0)/
      DATA C(1), C(2), C(3), C(4), C(5), C(6), C(7), C(8), C(9), C(10),
     1     C(11), C(12), C(13), C(14), C(15), C(16), C(17), C(18),
     2     C(19), C(20), C(21), C(22), C(23), C(24)/
     3     1.00000000000000000E+00,    -2.08333333333333333E-01,
     4     1.25000000000000000E-01,     3.34201388888888889E-01,
     5    -4.01041666666666667E-01,     7.03125000000000000E-02,
     6    -1.02581259645061728E+00,     1.84646267361111111E+00,
     7    -8.91210937500000000E-01,     7.32421875000000000E-02,
     8     4.66958442342624743E+00,    -1.12070026162229938E+01,
     9     8.78912353515625000E+00,    -2.36408691406250000E+00,
     A     1.12152099609375000E-01,    -2.82120725582002449E+01,
     B     8.46362176746007346E+01,    -9.18182415432400174E+01,
     C     4.25349987453884549E+01,    -7.36879435947963170E+00,
     D     2.27108001708984375E-01,     2.12570130039217123E+02,
     E    -7.65252468141181642E+02,     1.05999045252799988E+03/
      DATA C(25), C(26), C(27), C(28), C(29), C(30), C(31), C(32),
     1     C(33), C(34), C(35), C(36), C(37), C(38), C(39), C(40),
     2     C(41), C(42), C(43), C(44), C(45), C(46), C(47), C(48)/
     3    -6.99579627376132541E+02,     2.18190511744211590E+02,
     4    -2.64914304869515555E+01,     5.72501420974731445E-01,
     5    -1.91945766231840700E+03,     8.06172218173730938E+03,
     6    -1.35865500064341374E+04,     1.16553933368645332E+04,
     7    -5.30564697861340311E+03,     1.20090291321635246E+03,
     8    -1.08090919788394656E+02,     1.72772750258445740E+00,
     9     2.02042913309661486E+04,    -9.69805983886375135E+04,
     A     1.92547001232531532E+05,    -2.03400177280415534E+05,
     B     1.22200464983017460E+05,    -4.11926549688975513E+04,
     C     7.10951430248936372E+03,    -4.93915304773088012E+02,
     D     6.07404200127348304E+00,    -2.42919187900551333E+05,
     E     1.31176361466297720E+06,    -2.99801591853810675E+06/
      DATA C(49), C(50), C(51), C(52), C(53), C(54), C(55), C(56),
     1     C(57), C(58), C(59), C(60), C(61), C(62), C(63), C(64),
     2     C(65), C(66), C(67), C(68), C(69), C(70), C(71), C(72)/
```

```
     3   3.76327129765640400E+06,   -2.81356322658653411E+06,
     4   1.26836527332162478E+06,   -3.31645172484563578E+05,
     5   4.52187689813627263E+04,   -2.49983048181120962E+03,
     6   2.43805296995560639E+01,    3.28446985307203782E+06,
     7  -1.97068191184322269E+07,    5.09526024926646422E+07,
     8  -7.41051482115326577E+07,    6.63445122747290267E+07,
     9  -3.75671766607633513E+07,    1.32887671664218183E+07,
     A  -2.78561812808645469E+06,    3.08186404612662398E+05,
     B  -1.38860897537170405E+04,    1.10017140269246738E+02,
     C  -4.93292536645099620E+07,    3.25573074185765749E+08,
     D  -9.39462359681578403E+08,    1.55359689957058006E+09,
     E  -1.62108055210833708E+09,    1.10684281682301447E+09/
      DATA C(73), C(74), C(75), C(76), C(77), C(78), C(79), C(80),
     1   C(81), C(82), C(83), C(84), C(85), C(86), C(87), C(88),
     2   C(89), C(90), C(91), C(92), C(93), C(94), C(95), C(96)/
     3  -4.95889784275030309E+08,    1.42062907797533095E+08,
     4  -2.44740627257387285E+07,    2.24376817792244943E+06,
     5  -8.40054336030240853E+04,    5.51335896122020586E+02,
     6   8.14789096118312115E+08,   -5.86648149205184723E+09,
     7   1.86882075092958249E+10,   -3.46320433881587779E+10,
     8   4.12801855797539740E+10,   -3.30265997498007231E+10,
     9   1.79542137311556001E+10,   -6.56329379261928433E+09,
     A   1.55927986487925751E+09,   -2.25105661889415278E+08,
     B   1.73951075539781645E+07,   -5.49842327572288687E+05,
     C   3.03809051092238427E+03,   -1.46792612476956167E+10,
     D   1.14498237732025810E+11,   -3.99096175224466498E+11,
     E   8.19218669548577329E+11,   -1.09837515608122331E+12/
      DATA C(97), C(98), C(99), C(100), C(101), C(102), C(103), C(104),
     1   C(105), C(106), C(107), C(108), C(109), C(110), C(111),
     2   C(112), C(113), C(114), C(115), C(116), C(117), C(118)/
     3   1.00815810686538209E+12,   -6.45364869245376503E+11,
     4   2.87900649906150589E+11,   -8.78670721780232657E+10,
     5   1.76347306068349694E+10,   -2.16716498322379509E+09,
     6   1.43157876718888981E+08,   -3.87183344257261262E+06,
     7   1.82577554742931747E+04,    2.86464035717679043E+11,
     8  -2.40629790002850396E+12,    9.10934118523989896E+12,
     9  -2.05168994109344374E+13,    3.05651255199353206E+13,
     A  -3.16670885847851584E+13,    2.33483640445818409E+13,
     B  -1.23204913055982872E+13,    4.61272578084913197E+12,
     C  -1.19655288019618160E+12,    2.05914503232410016E+11,
     D  -2.18229277575292237E+10,    1.24700929351271032E+09/
      DATA C(119), C(120)/
     1  -2.91883881222208134E+07,    1.18838426256783253E+05/
C
      IF (INIT.NE.0) GO TO 40
C-----------------------------------------------------------------------
C     INITIALIZE ALL VARIABLES
C-----------------------------------------------------------------------
      RFN = 1.0E0/FNU
      CRFN = CMPLX(RFN,0.0E0)
C     T = ZR*CRFN
C-----------------------------------------------------------------------
C     OVERFLOW TEST (ZR/FNU TOO SMALL)
C-----------------------------------------------------------------------
      TSTR = REAL(ZR)
      TSTI = AIMAG(ZR)
      TEST = R1MACH(1)*1.0E+3
      AC = FNU*TEST
      IF (ABS(TSTR).GT.AC .OR. ABS(TSTI).GT.AC) GO TO 15
      AC = 2.0E0*ABS(ALOG(TEST))+FNU
      ZETA1 = CMPLX(AC,0.0E0)
```

```
      ZETA2 = CMPLX(FNU,0.0E0)
      PHI=CONE
      RETURN
  15 CONTINUE
      T=ZR*CRFN
      S = CONE + T*T
      SR = CSQRT(S)
      CFN = CMPLX(FNU,0.0E0)
      ZN = (CONE+SR)/T
      ZETA1 = CFN*CLOG(ZN)
      ZETA2 = CFN*SR
      T = CONE/SR
      SR = T*CRFN
      CWRK(16) = CSQRT(SR)
      PHI = CWRK(16)*CON(IKFLG)
      IF (IPMTR.NE.0) RETURN
      T2 = CONE/S
      CWRK(1) = CONE
      CRFN = CONE
      AC = 1.0E0
      L = 1
      DO 20 K=2,15
        S = CZERO
        DO 10 J=1,K
          L = L + 1
          S = S*T2 + CMPLX(C(L),0.0E0)
  10      CONTINUE
        CRFN = CRFN*SR
        CWRK(K) = CRFN*S
        AC = AC*RFN
        TSTR = REAL(CWRK(K))
        TSTI = AIMAG(CWRK(K))
        TEST = ABS(TSTR) + ABS(TSTI)
        IF (AC.LT.TOL .AND. TEST.LT.TOL) GO TO 30
  20 CONTINUE
      K = 15
  30 CONTINUE
      INIT = K
  40 CONTINUE
      IF (IKFLG.EQ.2) GO TO 60
C---------------------------------------------------------------
C     COMPUTE SUM FOR THE I FUNCTION
C---------------------------------------------------------------
      S = CZERO
      DO 50 I=1,INIT
        S = S + CWRK(I)
  50 CONTINUE
      SUM = S
      PHI = CWRK(16)*CON(1)
      RETURN
  60 CONTINUE
C---------------------------------------------------------------
C     COMPUTE SUM FOR THE K FUNCTION
C---------------------------------------------------------------
      S = CZERO
      T = CONE
      DO 70 I=1,INIT
        S = S + T*CWRK(I)
        T = -T
  70 CONTINUE
      SUM = S
```

```
      PHI = CWRK(16)*CON(2)
      RETURN
      END
      SUBROUTINE CUNK1(Z, FNU, KODE, MR, N, Y, NZ, TOL, ELIM, ALIM)
C***BEGIN PROLOGUE  CUNK1
C***REFER TO  CBESK
C
C     CUNK1 COMPUTES K(FNU,Z) AND ITS ANALYTIC CONTINUATION FROM THE
C     RIGHT HALF PLANE TO THE LEFT HALF PLANE BY MEANS OF THE
C     UNIFORM ASYMPTOTIC EXPANSION.
C     MR INDICATES THE DIRECTION OF ROTATION FOR ANALYTIC CONTINUATION.
C     NZ=-1 MEANS AN OVERFLOW WILL OCCUR
C
C***ROUTINES CALLED  CS1S2,CUCHK,CUNIK,R1MACH
C***END PROLOGUE  CUNK1
      COMPLEX CFN, CK, CONE, CRSC, CS, CSCL, CSGN, CSPN, CSR, CSS,
     * CWRK, CY, CZERO, C1, C2, PHI,  RZ, SUM,  S1, S2, Y, Z,
     * ZETA1,  ZETA2,  ZR, PHID, ZETA1D, ZETA2D, SUMD
      REAL ALIM, ANG, APHI, ASC, ASCLE, BRY, CPN, C2I, C2M, C2R, ELIM,
     * FMR, FN, FNF, FNU, PI, RS1, SGN, SPN, TOL, X, R1MACH
      INTEGER I, IB, IFLAG, IFN, IL, INIT, INU, IUF, K, KDFLG, KFLAG,
     * KK, KODE, MR, N, NW, NZ, J, IPARD, INITD, IC
      DIMENSION BRY(3), INIT(2), Y(N), SUM(2), PHI(2), ZETA1(2),
     * ZETA2(2), CY(2), CWRK(16,3), CSS(3), CSR(3)
      DATA CZERO, CONE / (0.0E0,0.0E0) , (1.0E0,0.0E0) /
      DATA PI / 3.14159265358979324E0 /
C
      KDFLG = 1
      NZ = 0
C-----------------------------------------------------------------------
C     EXP(-ALIM)=EXP(-ELIM)/TOL=APPROX. ONE PRECISION GREATER THAN
C     THE UNDERFLOW LIMIT
C-----------------------------------------------------------------------
      CSCL = CMPLX(1.0E0/TOL,0.0E0)
      CRSC = CMPLX(TOL,0.0E0)
      CSS(1) = CSCL
      CSS(2) = CONE
      CSS(3) = CRSC
      CSR(1) = CRSC
      CSR(2) = CONE
      CSR(3) = CSCL
      BRY(1) = 1.0E+3*R1MACH(1)/TOL
      BRY(2) = 1.0E0/BRY(1)
      BRY(3) = R1MACH(2)
      X = REAL(Z)
      ZR = Z
      IF (X.LT.0.0E0) ZR = -Z
      J=2
      DO 70 I=1,N
C-----------------------------------------------------------------------
C     J FLIP FLOPS BETWEEN 1 AND 2 IN J = 3 - J
C-----------------------------------------------------------------------
        J = 3 - J
        FN = FNU + FLOAT(I-1)
        INIT(J) = 0
        CALL CUNIK(ZR, FN, 2, 0, TOL, INIT(J), PHI(J), ZETA1(J),
     *   ZETA2(J), SUM(J), CWRK(1,J))
        IF (KODE.EQ.1) GO TO 20
        CFN = CMPLX(FN,0.0E0)
        S1 = ZETA1(J) - CFN*(CFN/(ZR+ZETA2(J)))
        GO TO 30
```

```
   20  CONTINUE
       S1 = ZETA1(J) - ZETA2(J)
   30  CONTINUE
C-----------------------------------------------------------------
C    TEST FOR UNDERFLOW AND OVERFLOW
C-----------------------------------------------------------------
       RS1 = REAL(S1)
       IF (ABS(RS1).GT.ELIM) GO TO 60
       IF (KDFLG.EQ.1) KFLAG = 2
       IF (ABS(RS1).LT.ALIM) GO TO 40
C-----------------------------------------------------------------
C    REFINE  TEST AND SCALE
C-----------------------------------------------------------------
       APHI = CABS(PHI(J))
       RS1 = RS1 + ALOG(APHI)
       IF (ABS(RS1).GT.ELIM) GO TO 60
       IF (KDFLG.EQ.1) KFLAG = 1
       IF (RS1.LT.0.0E0) GO TO 40
       IF (KDFLG.EQ.1) KFLAG = 3
   40  CONTINUE
C-----------------------------------------------------------------
C    SCALE S1 TO KEEP INTERMEDIATE ARITHMETIC ON SCALE NEAR
C    EXPONENT EXTREMES
C-----------------------------------------------------------------
       S2 = PHI(J)*SUM(J)
       C2R = REAL(S1)
       C2I = AIMAG(S1)
       C2M = EXP(C2R)*REAL(CSS(KFLAG))
       S1 = CMPLX(C2M,0.0E0)*CMPLX(COS(C2I),SIN(C2I))
       S2 = S2*S1
       IF (KFLAG.NE.1) GO TO 50
       CALL CUCHK(S2, NW, BRY(1), TOL)
       IF (NW.NE.0) GO TO 60
   50  CONTINUE
       CY(KDFLG) = S2
       Y(I) = S2*CSR(KFLAG)
       IF (KDFLG.EQ.2) GO TO 75
       KDFLG = 2
       GO TO 70
   60  CONTINUE
       IF (RS1.GT.0.0E0) GO TO 290
C-----------------------------------------------------------------
C    FOR X.LT.0.0, THE I FUNCTION TO BE ADDED WILL OVERFLOW
C-----------------------------------------------------------------
       IF (X.LT.0.0E0) GO TO 290
       KDFLG = 1
       Y(I) = CZERO
       NZ=NZ+1
       IF (I.EQ.1) GO TO 70
       IF (Y(I-1).EQ.CZERO) GO TO 70
       Y(I-1) = CZERO
       NZ=NZ+1
   70 CONTINUE
     I=N
   75 CONTINUE
     RZ = CMPLX(2.0E0,0.0E0)/ZR
     CK = CMPLX(FN,0.0E0)*RZ
     IB = I+1
     IF (N.LT.IB) GO TO 160
C-----------------------------------------------------------------
```

```
C     TEST LAST MEMBER FOR UNDERFLOW AND OVERFLOW, SET SEQUENCE TO
ZERO
C     ON UNDERFLOW
C-----------------------------------------------------------------------
      FN = FNU+FLOAT(N-1)
      IPARD = 1
      IF (MR.NE.0) IPARD = 0
      INITD = 0
      CALL CUNIK(ZR,FN,2,IPARD,TOL,INITD,PHID,ZETA1D,ZETA2D,SUMD,
     *CWRK(1,3))
      IF (KODE.EQ.1) GO TO 80
      CFN=CMPLX(FN,0.0E0)
      S1=ZETA1D-CFN*(CFN/(ZR+ZETA2D))
      GO TO 90
   80 CONTINUE
      S1=ZETA1D-ZETA2D
   90 CONTINUE
      RS1=REAL(S1)
      IF (ABS(RS1).GT.ELIM) GO TO 95
      IF (ABS(RS1).LT.ALIM) GO TO 100
C-----------------------------------------------------------------------
C     REFINE ESTIMATE AND TEST
C-----------------------------------------------------------------------
      APHI=CABS(PHID)
      RS1=RS1+ALOG(APHI)
      IF (ABS(RS1).LT.ELIM) GO TO 100
   95 CONTINUE
      IF (RS1.GT.0.0E0) GO TO 290
C-----------------------------------------------------------------------
C     FOR X.LT.0.0, THE I FUNCTION TO BE ADDED WILL OVERFLOW
C-----------------------------------------------------------------------
      IF (X.LT.0.0E0) GO TO 290
      NZ=N
      DO 96 I=1,N
        Y(I) = CZERO
   96 CONTINUE
      RETURN
  100 CONTINUE
C-----------------------------------------------------------------------
C     RECUR FORWARD FOR REMAINDER OF THE SEQUENCE
C-----------------------------------------------------------------------
      S1 = CY(1)
      S2 = CY(2)
      C1 = CSR(KFLAG)
      ASCLE = BRY(KFLAG)
      DO 120 I=IB,N
        C2 = S2
        S2 = CK*S2 + S1
        S1 = C2
        CK = CK + RZ
        C2 = S2*C1
        Y(I) = C2
        IF (KFLAG.GE.3) GO TO 120
        C2R = REAL(C2)
        C2I = AIMAG(C2)
        C2R = ABS(C2R)
        C2I = ABS(C2I)
        C2M = AMAX1(C2R,C2I)
        IF (C2M.LE.ASCLE) GO TO 120
        KFLAG = KFLAG + 1
        ASCLE = BRY(KFLAG)
```

```
          S1 = S1*C1
          S2 = C2
          S1 = S1*CSS(KFLAG)
          S2 = S2*CSS(KFLAG)
          C1 = CSR(KFLAG)
  120 CONTINUE
  160 CONTINUE
        IF (MR.EQ.0) RETURN
C-----------------------------------------------------------------------
C     ANALYTIC CONTINUATION FOR RE(Z).LT.0.0E0
C-----------------------------------------------------------------------
        NZ = 0
        FMR = FLOAT(MR)
        SGN = -SIGN(PI,FMR)
C-----------------------------------------------------------------------
C     CSPN AND CSGN ARE COEFF OF K AND I FUNCIONS RESP.
C-----------------------------------------------------------------------
        CSGN = CMPLX(0.0E0,SGN)
        INU = INT(FNU)
        FNF = FNU - FLOAT(INU)
        IFN = INU + N - 1
        ANG = FNF*SGN
        CPN = COS(ANG)
        SPN = SIN(ANG)
        CSPN = CMPLX(CPN,SPN)
        IF (MOD(IFN,2).EQ.1) CSPN = -CSPN
        ASC = BRY(1)
        KK = N
        IUF = 0
        KDFLG = 1
        IB = IB-1
        IC = IB-1
        DO 260 K=1,N
          FN = FNU + FLOAT(KK-1)
C-----------------------------------------------------------------------
C     LOGIC TO SORT OUT CASES WHOSE PARAMETERS WERE SET FOR THE K
C     FUNCTION ABOVE
C-----------------------------------------------------------------------
          M=3
          IF (N.GT.2) GO TO 175
  170     CONTINUE
          INITD = INIT(J)
          PHID = PHI(J)
          ZETA1D = ZETA1(J)
          ZETA2D = ZETA2(J)
          SUMD = SUM(J)
          M = J
          J = 3 - J
          GO TO 180
  175     CONTINUE
          IF ((KK.EQ.N).AND.(IB.LT.N)) GO TO 180
          IF ((KK.EQ.IB).OR.(KK.EQ.IC)) GO TO 170
          INITD = 0
  180     CONTINUE
          CALL CUNIK(ZR, FN, 1, 0, TOL, INITD, PHID, ZETA1D,
     *     ZETA2D, SUMD, CWRK(1,M))
          IF (KODE.EQ.1) GO TO 190
          CFN = CMPLX(FN,0.0E0)
          S1 = -ZETA1D + CFN*(CFN/(ZR+ZETA2D))
          GO TO 200
  190     CONTINUE
```

```
      S1 = -ZETA1D + ZETA2D
  200   CONTINUE
C-----------------------------------------------------------------
C     TEST FOR UNDERFLOW AND OVERFLOW
C-----------------------------------------------------------------
      RS1 = REAL(S1)
      IF (ABS(RS1).GT.ELIM) GO TO 250
      IF (KDFLG.EQ.1) IFLAG = 2
      IF (ABS(RS1).LT.ALIM) GO TO 210
C-----------------------------------------------------------------
C     REFINE  TEST AND SCALE
C-----------------------------------------------------------------
      APHI = CABS(PHID)
      RS1 = RS1 + ALOG(APHI)
      IF (ABS(RS1).GT.ELIM) GO TO 250
      IF (KDFLG.EQ.1) IFLAG = 1
      IF (RS1.LT.0.0E0) GO TO 210
      IF (KDFLG.EQ.1) IFLAG = 3
  210   CONTINUE
      S2 = CSGN*PHID*SUMD
      C2R = REAL(S1)
      C2I = AIMAG(S1)
      C2M = EXP(C2R)*REAL(CSS(IFLAG))
      S1 = CMPLX(C2M,0.0E0)*CMPLX(COS(C2I),SIN(C2I))
      S2 = S2*S1
      IF (IFLAG.NE.1) GO TO 220
      CALL CUCHK(S2, NW, BRY(1), TOL)
      IF (NW.NE.0) S2 = CMPLX(0.0E0,0.0E0)
  220   CONTINUE
      CY(KDFLG) = S2
      C2 = S2
      S2 = S2*CSR(IFLAG)
C-----------------------------------------------------------------
C     ADD I AND K FUNCTIONS, K SEQUENCE IN Y(I), I=1,N
C-----------------------------------------------------------------
      S1 = Y(KK)
      IF (KODE.EQ.1) GO TO 240
      CALL CS1S2(ZR, S1, S2, NW, ASC, ALIM, IUF)
      NZ = NZ + NW
  240   CONTINUE
      Y(KK) = S1*CSPN + S2
      KK = KK - 1
      CSPN = -CSPN
      IF (C2.NE.CZERO) GO TO 245
      KDFLG = 1
      GO TO 260
  245   CONTINUE
      IF (KDFLG.EQ.2) GO TO 265
      KDFLG = 2
      GO TO 260
  250   CONTINUE
      IF (RS1.GT.0.0E0) GO TO 290
      S2 = CZERO
      GO TO 220
  260 CONTINUE
    K = N
  265 CONTINUE
    IL = N - K
    IF (IL.EQ.0) RETURN
C-----------------------------------------------------------------
C     RECUR BACKWARD FOR REMAINDER OF I SEQUENCE AND ADD IN THE
```

```
C     K FUNCTIONS, SCALING THE I SEQUENCE DURING RECURRENCE TO KEEP
C     INTERMEDIATE ARITHMETIC ON SCALE NEAR EXPONENT EXTREMES.
C-----------------------------------------------------------------------
      S1 = CY(1)
      S2 = CY(2)
      CS = CSR(IFLAG)
      ASCLE = BRY(IFLAG)
      FN = FLOAT(INU+IL)
      DO 280 I=1,IL
        C2 = S2
        S2 = S1 + CMPLX(FN+FNF,0.0E0)*RZ*S2
        S1 = C2
        FN = FN - 1.0E0
        C2 = S2*CS
        CK = C2
        C1 = Y(KK)
        IF (KODE.EQ.1) GO TO 270
        CALL CS1S2(ZR, C1, C2, NW, ASC, ALIM, IUF)
        NZ = NZ + NW
  270   CONTINUE
        Y(KK) = C1*CSPN + C2
        KK = KK - 1
        CSPN = -CSPN
        IF (IFLAG.GE.3) GO TO 280
        C2R = REAL(CK)
        C2I = AIMAG(CK)
        C2R = ABS(C2R)
        C2I = ABS(C2I)
        C2M = AMAX1(C2R,C2I)
        IF (C2M.LE.ASCLE) GO TO 280
        IFLAG = IFLAG + 1
        ASCLE = BRY(IFLAG)
        S1 = S1*CS
        S2 = CK
        S1 = S1*CSS(IFLAG)
        S2 = S2*CSS(IFLAG)
        CS = CSR(IFLAG)
  280 CONTINUE
      RETURN
  290 CONTINUE
      NZ = -1
      RETURN
      END
      SUBROUTINE CUNK2(Z, FNU, KODE, MR, N, Y, NZ, TOL, ELIM, ALIM)
C***BEGIN PROLOGUE  CUNK2
C***REFER TO  CBESK
C
C     CUNK2 COMPUTES K(FNU,Z) AND ITS ANALYTIC CONTINUATION FROM THE
C     RIGHT HALF PLANE TO THE LEFT HALF PLANE BY MEANS OF THE
C     UNIFORM ASYMPTOTIC EXPANSIONS FOR H(KIND,FNU,ZN) AND J(FNU,ZN)
C     WHERE ZN IS IN THE RIGHT HALF PLANE, KIND=(3-MR)/2, MR=+1 OR
C     -1. HERE ZN=ZR*I OR -ZR*I WHERE ZR=Z IF Z IS IN THE RIGHT
C     HALF PLANE OR ZR=-Z IF Z IS IN THE LEFT HALF PLANE. MR INDIC-
C     ATES THE DIRECTION OF ROTATION FOR ANALYTIC CONTINUATION.
C     NZ=-1 MEANS AN OVERFLOW WILL OCCUR
C
C***ROUTINES CALLED  CAIRY,CS1S2,CUCHK,CUNHJ,R1MACH
C***END PROLOGUE  CUNK2
      COMPLEX AI, ARG, ASUM, BSUM, CFN, CI, CIP,
     * CK, CONE, CRSC, CR1, CR2, CS, CSCL, CSGN, CSPN, CSR, CSS, CY,
     * CZERO, C1, C2, DAI, PHI,  RZ, S1, S2, Y, Z, ZB, ZETA1,
```

```
     * ZETA2, ZN, ZR, PHID, ARGD, ZETA1D, ZETA2D, ASUMD, BSUMD
      REAL AARG, AIC, ALIM, ANG, APHI, ASC, ASCLE, BRY, CAR, CPN, C2I,
     * C2M, C2R, ELIM, FMR, FN, FNF, FNU, HPI, PI, RS1, SAR, SGN, SPN,
     * TOL, X, YY, R1MACH
      INTEGER I, IB, IFLAG, IFN, IL, IN, INU, IUF, K, KDFLG, KFLAG, KK,
     * KODE, MR, N, NAI, NDAI, NW, NZ, IDUM, J, IPARD, IC
      DIMENSION BRY(3), Y(N), ASUM(2), BSUM(2), PHI(2), ARG(2),
     * ZETA1(2), ZETA2(2), CY(2), CIP(4), CSS(3), CSR(3)
      DATA CZERO, CONE, CI, CR1, CR2 /
     1        (0.0E0,0.0E0),(1.0E0,0.0E0),(0.0E0,1.0E0),
     1(1.0E0,1.73205080756887729E0),(-0.5E0,-8.66025403784438647E-01)/
      DATA HPI, PI, AIC /
     1    1.57079632679489662E+00,    3.14159265358979324E+00,
     1    1.26551212348464539E+00/
      DATA CIP(1),CIP(2),CIP(3),CIP(4)/
     1 (1.0E0,0.0E0), (0.0E0,-1.0E0), (-1.0E0,0.0E0), (0.0E0,1.0E0)/
C
      KDFLG = 1
      NZ = 0
C-----------------------------------------------------------------------
C     EXP(-ALIM)=EXP(-ELIM)/TOL=APPROX. ONE PRECISION GREATER THAN
C     THE UNDERFLOW LIMIT
C-----------------------------------------------------------------------
      CSCL = CMPLX(1.0E0/TOL,0.0E0)
      CRSC = CMPLX(TOL,0.0E0)
      CSS(1) = CSCL
      CSS(2) = CONE
      CSS(3) = CRSC
      CSR(1) = CRSC
      CSR(2) = CONE
      CSR(3) = CSCL
      BRY(1) = 1.0E+3*R1MACH(1)/TOL
      BRY(2) = 1.0E0/BRY(1)
      BRY(3) = R1MACH(2)
      X = REAL(Z)
      ZR = Z
      IF (X.LT.0.0E0) ZR = -Z
      YY = AIMAG(ZR)
      ZN = -ZR*CI
      ZB = ZR
      INU = INT(FNU)
      FNF = FNU - FLOAT(INU)
      ANG = -HPI*FNF
      CAR = COS(ANG)
      SAR = SIN(ANG)
      CPN = -HPI*CAR
      SPN = -HPI*SAR
      C2 = CMPLX(-SPN,CPN)
      KK = MOD(INU,4) + 1
      CS = CR1*C2*CIP(KK)
      IF (YY.GT.0.0E0) GO TO 10
      ZN = CONJG(-ZN)
      ZB = CONJG(ZB)
   10 CONTINUE
C-----------------------------------------------------------------------
C     K(FNU,Z) IS COMPUTED FROM H(2,FNU,-I*Z) WHERE Z IS IN THE FIRST
C     QUADRANT. FOURTH QUADRANT VALUES (YY.LE.0.0E0) ARE COMPUTED BY
C     CONJUGATION SINCE THE K FUNCTION IS REAL ON THE POSITIVE REAL AXIS
C-----------------------------------------------------------------------
      J = 2
      DO 70 I=1,N
```

```
C-----------------------------------------------------------------------
C     J FLIP FLOPS BETWEEN 1 AND 2 IN J = 3 - J
C-----------------------------------------------------------------------
      J = 3 - J
      FN = FNU + FLOAT(I-1)
      CALL CUNHJ(ZN, FN, 0, TOL, PHI(J), ARG(J), ZETA1(J), ZETA2(J),
     *  ASUM(J), BSUM(J))
      IF (KODE.EQ.1) GO TO 20
      CFN = CMPLX(FN,0.0E0)
      S1 = ZETA1(J) - CFN*(CFN/(ZB+ZETA2(J)))
      GO TO 30
   20 CONTINUE
      S1 = ZETA1(J) - ZETA2(J)
   30 CONTINUE
C-----------------------------------------------------------------------
C     TEST FOR UNDERFLOW AND OVERFLOW
C-----------------------------------------------------------------------
      RS1 = REAL(S1)
      IF (ABS(RS1).GT.ELIM) GO TO 60
      IF (KDFLG.EQ.1) KFLAG = 2
      IF (ABS(RS1).LT.ALIM) GO TO 40
C-----------------------------------------------------------------------
C     REFINE  TEST AND SCALE
C-----------------------------------------------------------------------
      APHI = CABS(PHI(J))
      AARG = CABS(ARG(J))
      RS1 = RS1 + ALOG(APHI) - 0.25E0*ALOG(AARG) - AIC
      IF (ABS(RS1).GT.ELIM) GO TO 60
      IF (KDFLG.EQ.1) KFLAG = 1
      IF (RS1.LT.0.0E0) GO TO 40
      IF (KDFLG.EQ.1) KFLAG = 3
   40 CONTINUE
C-----------------------------------------------------------------------
C     SCALE S1 TO KEEP INTERMEDIATE ARITHMETIC ON SCALE NEAR
C     EXPONENT EXTREMES
C-----------------------------------------------------------------------
      C2 = ARG(J)*CR2
      CALL CAIRY(C2, 0, 2, AI, NAI, IDUM)
      CALL CAIRY(C2, 1, 2, DAI, NDAI, IDUM)
      S2 = CS*PHI(J)*(AI*ASUM(J)+CR2*DAI*BSUM(J))
      C2R = REAL(S1)
      C2I = AIMAG(S1)
      C2M = EXP(C2R)*REAL(CSS(KFLAG))
      S1 = CMPLX(C2M,0.0E0)*CMPLX(COS(C2I),SIN(C2I))
      S2 = S2*S1
      IF (KFLAG.NE.1) GO TO 50
      CALL CUCHK(S2, NW, BRY(1), TOL)
      IF (NW.NE.0) GO TO 60
   50 CONTINUE
      IF (YY.LE.0.0E0) S2 = CONJG(S2)
      CY(KDFLG) = S2
      Y(I) = S2*CSR(KFLAG)
      CS = -CI*CS
      IF (KDFLG.EQ.2) GO TO 75
      KDFLG = 2
      GO TO 70
   60 CONTINUE
      IF (RS1.GT.0.0E0) GO TO 300
C-----------------------------------------------------------------------
C     FOR X.LT.0.0, THE I FUNCTION TO BE ADDED WILL OVERFLOW
C-----------------------------------------------------------------------
```

```
       IF (X.LT.0.0E0) GO TO 300
       KDFLG = 1
       Y(I) = CZERO
       CS = -CI*CS
       NZ=NZ+1
       IF (I.EQ.1) GO TO 70
       IF (Y(I-1).EQ.CZERO) GO TO 70
       Y(I-1) = CZERO
       NZ=NZ+1
   70 CONTINUE
      I=N
   75 CONTINUE
      RZ = CMPLX(2.0E0,0.0E0)/ZR
      CK = CMPLX(FN,0.0E0)*RZ
      IB = I + 1
      IF (N.LT.IB) GO TO 170
C-----------------------------------------------------------------------
C     TEST LAST MEMBER FOR UNDERFLOW AND OVERFLOW, SET SEQUENCE TO
ZERO
C     ON UNDERFLOW
C-----------------------------------------------------------------------
      FN = FNU+FLOAT(N-1)
      IPARD = 1
      IF (MR.NE.0) IPARD = 0
      CALL CUNHJ(ZN,FN,IPARD,TOL,PHID,ARGD,ZETA1D,ZETA2D,ASUMD,BSUMD)
      IF (KODE.EQ.1) GO TO 80
      CFN=CMPLX(FN,0.0E0)
      S1=ZETA1D-CFN*(CFN/(ZB+ZETA2D))
      GO TO 90
   80 CONTINUE
      S1=ZETA1D-ZETA2D
   90 CONTINUE
      RS1=REAL(S1)
      IF (ABS(RS1).GT.ELIM) GO TO 95
      IF (ABS(RS1).LT.ALIM) GO TO 100
C-----------------------------------------------------------------------
C     REFINE ESTIMATE AND TEST
C-----------------------------------------------------------------------
      APHI=CABS(PHID)
      AARG = CABS(ARGD)
      RS1=RS1+ALOG(APHI)-0.25E0*ALOG(AARG)-AIC
      IF (ABS(RS1).LT.ELIM) GO TO 100
   95 CONTINUE
      IF (RS1.GT.0.0E0) GO TO 300
C-----------------------------------------------------------------------
C     FOR X.LT.0.0, THE I FUNCTION TO BE ADDED WILL OVERFLOW
C-----------------------------------------------------------------------
      IF (X.LT.0.0E0) GO TO 300
      NZ=N
      DO 96 I=1,N
        Y(I) = CZERO
   96 CONTINUE
      RETURN
  100 CONTINUE
C-----------------------------------------------------------------------
C     SCALED FORWARD RECURRENCE FOR REMAINDER OF THE SEQUENCE
C-----------------------------------------------------------------------
      S1 = CY(1)
      S2 = CY(2)
      C1 = CSR(KFLAG)
      ASCLE = BRY(KFLAG)
```

```
      DO 120 I=IB,N
        C2 = S2
        S2 = CK*S2 + S1
        S1 = C2
        CK = CK + RZ
        C2 = S2*C1
        Y(I) = C2
        IF (KFLAG.GE.3) GO TO 120
        C2R = REAL(C2)
        C2I = AIMAG(C2)
        C2R = ABS(C2R)
        C2I = ABS(C2I)
        C2M = AMAX1(C2R,C2I)
        IF (C2M.LE.ASCLE) GO TO 120
        KFLAG = KFLAG + 1
        ASCLE = BRY(KFLAG)
        S1 = S1*C1
        S2 = C2
        S1 = S1*CSS(KFLAG)
        S2 = S2*CSS(KFLAG)
        C1 = CSR(KFLAG)
  120 CONTINUE
  170 CONTINUE
      IF (MR.EQ.0) RETURN
C-----------------------------------------------------------------------
C     ANALYTIC CONTINUATION FOR RE(Z).LT.0.0E0
C-----------------------------------------------------------------------
      NZ = 0
      FMR = FLOAT(MR)
      SGN = -SIGN(PI,FMR)
C-----------------------------------------------------------------------
C     CSPN AND CSGN ARE COEFF OF K AND I FUNCTIONS RESP.
C-----------------------------------------------------------------------
      CSGN = CMPLX(0.0E0,SGN)
      IF (YY.LE.0.0E0) CSGN = CONJG(CSGN)
      IFN = INU + N - 1
      ANG = FNF*SGN
      CPN = COS(ANG)
      SPN = SIN(ANG)
      CSPN = CMPLX(CPN,SPN)
      IF (MOD(IFN,2).EQ.1) CSPN = -CSPN
C-----------------------------------------------------------------------
C     CS=COEFF OF THE J FUNCTION TO GET THE I FUNCTION. I(FNU,Z) IS
C     COMPUTED FROM EXP(I*FNU*HPI)*J(FNU,-I*Z) WHERE Z IS IN THE FIRST
C     QUADRANT. FOURTH QUADRANT VALUES (YY.LE.0.0E0) ARE COMPUTED BY
C     CONJUGATION SINCE THE I FUNCTION IS REAL ON THE POSITIVE REAL AXIS
C-----------------------------------------------------------------------
      CS = CMPLX(CAR,-SAR)*CSGN
      IN = MOD(IFN,4) + 1
      C2 = CIP(IN)
      CS = CS*CONJG(C2)
      ASC = BRY(1)
      KK = N
      KDFLG = 1
      IB = IB-1
      IC = IB-1
      IUF = 0
      DO 270 K=1,N
C-----------------------------------------------------------------------
C     LOGIC TO SORT OUT CASES WHOSE PARAMETERS WERE SET FOR THE K
C     FUNCTION ABOVE
```

```
C-----------------------------------------------------------------------
      FN = FNU+FLOAT(KK-1)
      IF (N.GT.2) GO TO 180
  175 CONTINUE
      PHID = PHI(J)
      ARGD = ARG(J)
      ZETA1D = ZETA1(J)
      ZETA2D = ZETA2(J)
      ASUMD = ASUM(J)
      BSUMD = BSUM(J)
      J = 3 - J
      GO TO 190
  180 CONTINUE
      IF ((KK.EQ.N).AND.(IB.LT.N)) GO TO 190
      IF ((KK.EQ.IB).OR.(KK.EQ.IC)) GO TO 175
      CALL CUNHJ(ZN, FN, 0, TOL, PHID, ARGD, ZETA1D, ZETA2D,
     *  ASUMD, BSUMD)
  190 CONTINUE
      IF (KODE.EQ.1) GO TO 200
      CFN = CMPLX(FN,0.0E0)
      S1 = -ZETA1D + CFN*(CFN/(ZB+ZETA2D))
      GO TO 210
  200 CONTINUE
      S1 = -ZETA1D + ZETA2D
  210 CONTINUE
C-----------------------------------------------------------------------
C     TEST FOR UNDERFLOW AND OVERFLOW
C-----------------------------------------------------------------------
      RS1 = REAL(S1)
      IF (ABS(RS1).GT.ELIM) GO TO 260
      IF (KDFLG.EQ.1) IFLAG = 2
      IF (ABS(RS1).LT.ALIM) GO TO 220
C-----------------------------------------------------------------------
C     REFINE  TEST AND SCALE
C-----------------------------------------------------------------------
      APHI = CABS(PHID)
      AARG = CABS(ARGD)
      RS1 = RS1 + ALOG(APHI) - 0.25E0*ALOG(AARG) - AIC
      IF (ABS(RS1).GT.ELIM) GO TO 260
      IF (KDFLG.EQ.1) IFLAG = 1
      IF (RS1.LT.0.0E0) GO TO 220
      IF (KDFLG.EQ.1) IFLAG = 3
  220 CONTINUE
      CALL CAIRY(ARGD, 0, 2, AI, NAI, IDUM)
      CALL CAIRY(ARGD, 1, 2, DAI, NDAI, IDUM)
      S2 = CS*PHID*(AI*ASUMD+DAI*BSUMD)
      C2R = REAL(S1)
      C2I = AIMAG(S1)
      C2M = EXP(C2R)*REAL(CSS(IFLAG))
      S1 = CMPLX(C2M,0.0E0)*CMPLX(COS(C2I),SIN(C2I))
      S2 = S2*S1
      IF (IFLAG.NE.1) GO TO 230
      CALL CUCHK(S2, NW, BRY(1), TOL)
      IF (NW.NE.0) S2 = CMPLX(0.0E0,0.0E0)
  230 CONTINUE
      IF (YY.LE.0.0E0) S2 = CONJG(S2)
      CY(KDFLG) = S2
      C2 = S2
      S2 = S2*CSR(IFLAG)
C-----------------------------------------------------------------------
C     ADD I AND K FUNCTIONS, K SEQUENCE IN Y(I), I=1,N
```

```
C-------------------------------------------------------------
      S1 = Y(KK)
      IF (KODE.EQ.1) GO TO 250
      CALL CS1S2(ZR, S1, S2, NW, ASC, ALIM, IUF)
      NZ = NZ + NW
 250   CONTINUE
      Y(KK) = S1*CSPN + S2
      KK = KK - 1
      CSPN = -CSPN
      CS = -CS*CI
      IF (C2.NE.CZERO) GO TO 255
      KDFLG = 1
      GO TO 270
 255   CONTINUE
      IF (KDFLG.EQ.2) GO TO 275
      KDFLG = 2
      GO TO 270
 260   CONTINUE
      IF (RS1.GT.0.0E0) GO TO 300
      S2 = CZERO
      GO TO 230
 270 CONTINUE
    K = N
 275 CONTINUE
    IL = N-K
    IF (IL.EQ.0) RETURN
C-------------------------------------------------------------
C    RECUR BACKWARD FOR REMAINDER OF I SEQUENCE AND ADD IN THE
C    K FUNCTIONS, SCALING THE I SEQUENCE DURING RECURRENCE TO KEEP
C    INTERMEDIATE ARITHMETIC ON SCALE NEAR EXPONENT EXTREMES.
C-------------------------------------------------------------
    S1 = CY(1)
    S2 = CY(2)
    CS = CSR(IFLAG)
    ASCLE = BRY(IFLAG)
    FN = FLOAT(INU+IL)
    DO 290 I=1,IL
     C2 = S2
     S2 = S1 + CMPLX(FN+FNF,0.0E0)*RZ*S2
     S1 = C2
     FN = FN - 1.0E0
     C2 = S2*CS
     CK = C2
     C1 = Y(KK)
     IF (KODE.EQ.1) GO TO 280
     CALL CS1S2(ZR, C1, C2, NW, ASC, ALIM, IUF)
     NZ = NZ + NW
 280   CONTINUE
     Y(KK) = C1*CSPN + C2
     KK = KK - 1
     CSPN = -CSPN
     IF (IFLAG.GE.3) GO TO 290
     C2R = REAL(CK)
     C2I = AIMAG(CK)
     C2R = ABS(C2R)
     C2I = ABS(C2I)
     C2M = AMAX1(C2R,C2I)
     IF (C2M.LE.ASCLE) GO TO 290
     IFLAG = IFLAG + 1
     ASCLE = BRY(IFLAG)
     S1 = S1*CS
```

```
        S2 = CK
        S1 = S1*CSS(IFLAG)
        S2 = S2*CSS(IFLAG)
        CS = CSR(IFLAG)
  290 CONTINUE
      RETURN
  300 CONTINUE
      NZ = -1
      RETURN
      END
      SUBROUTINE CUOIK(Z, FNU, KODE, IKFLG, N, Y, NUF, TOL, ELIM, ALIM)
C***BEGIN PROLOGUE  CUOIK
C***REFER TO  CBESI,CBESK,CBESH
C
C     CUOIK COMPUTES THE LEADING TERMS OF THE UNIFORM ASYMPTOTIC
C     EXPANSIONS FOR THE I AND K FUNCTIONS AND COMPARES THEM
C     (IN LOGARITHMIC FORM) TO ALIM AND ELIM FOR OVER AND UNDERFLOW
C     WHERE ALIM.LT.ELIM. IF THE MAGNITUDE, BASED ON THE LEADING
C     EXPONENTIAL, IS LESS THAN ALIM OR GREATER THAN -ALIM, THEN
C     THE RESULT IS ON SCALE. IF NOT, THEN A REFINED TEST USING OTHER
C     MULTIPLIERS (IN LOGARITHMIC FORM) IS MADE BASED ON ELIM. HERE
C     EXP(-ELIM)=SMALLEST MACHINE NUMBER*1.0E+3 AND EXP(-ALIM)=
C     EXP(-ELIM)/TOL
C
C     IKFLG=1 MEANS THE I SEQUENCE IS TESTED
C          =2 MEANS THE K SEQUENCE IS TESTED
C     NUF = 0 MEANS THE LAST MEMBER OF THE SEQUENCE IS ON SCALE
C         =-1 MEANS AN OVERFLOW WOULD OCCUR
C     IKFLG=1 AND NUF.GT.0 MEANS THE LAST NUF Y VALUES WERE SET TO ZERO
C             THE FIRST N-NUF VALUES MUST BE SET BY ANOTHER ROUTINE
C     IKFLG=2 AND NUF.EQ.N MEANS ALL Y VALUES WERE SET TO ZERO
C     IKFLG=2 AND 0.LT.NUF.LT.N NOT CONSIDERED. Y MUST BE SET BY
C             ANOTHER ROUTINE
C
C***ROUTINES CALLED  CUCHK,CUNHJ,CUNIK,R1MACH
C***END PROLOGUE  CUOIK
      COMPLEX ARG, ASUM, BSUM, CWRK, CZ, CZERO, PHI, SUM, Y, Z, ZB,
     * ZETA1, ZETA2, ZN, ZR
      REAL AARG, AIC, ALIM, APHI, ASCLE, AX, AY, ELIM, FNN, FNU, GNN,
     * GNU, RCZ, TOL, X, YY
      INTEGER I, IFORM, IKFLG, INIT, KODE, N, NN, NUF, NW
      DIMENSION Y(N), CWRK(16)
      DATA CZERO / (0.0E0,0.0E0) /
      DATA AIC / 1.265512123484645396E+00 /
      NUF = 0
      NN = N
      X = REAL(Z)
      ZR = Z
      IF (X.LT.0.0E0) ZR = -Z
      ZB = ZR
      YY = AIMAG(ZR)
      AX = ABS(X)*1.7321E0
      AY = ABS(YY)
      IFORM = 1
      IF (AY.GT.AX) IFORM = 2
      GNU = AMAX1(FNU,1.0E0)
      IF (IKFLG.EQ.1) GO TO 10
      FNN = FLOAT(NN)
      GNN = FNU + FNN - 1.0E0
      GNU = AMAX1(GNN,FNN)
   10 CONTINUE
```

```
C-----------------------------------------------------------------
C     ONLY THE MAGNITUDE OF ARG AND PHI ARE NEEDED ALONG WITH THE
C     REAL PARTS OF ZETA1, ZETA2 AND ZB. NO ATTEMPT IS MADE TO GET
C     THE SIGN OF THE IMAGINARY PART CORRECT.
C-----------------------------------------------------------------
      IF (IFORM.EQ.2) GO TO 20
      INIT = 0
      CALL CUNIK(ZR, GNU, IKFLG, 1, TOL, INIT, PHI, ZETA1, ZETA2, SUM,
     * CWRK)
      CZ = -ZETA1 + ZETA2
      GO TO 40
   20 CONTINUE
      ZN = -ZR*CMPLX(0.0E0,1.0E0)
      IF (YY.GT.0.0E0) GO TO 30
      ZN = CONJG(-ZN)
   30 CONTINUE
      CALL CUNHJ(ZN, GNU, 1, TOL, PHI, ARG, ZETA1, ZETA2, ASUM, BSUM)
      CZ = -ZETA1 + ZETA2
      AARG = CABS(ARG)
   40 CONTINUE
      IF (KODE.EQ.2) CZ = CZ - ZB
      IF (IKFLG.EQ.2) CZ = -CZ
      APHI = CABS(PHI)
      RCZ = REAL(CZ)
C-----------------------------------------------------------------
C     OVERFLOW TEST
C-----------------------------------------------------------------
      IF (RCZ.GT.ELIM) GO TO 170
      IF (RCZ.LT.ALIM) GO TO 50
      RCZ = RCZ + ALOG(APHI)
      IF (IFORM.EQ.2) RCZ = RCZ - 0.25E0*ALOG(AARG) - AIC
      IF (RCZ.GT.ELIM) GO TO 170
      GO TO 100
   50 CONTINUE
C-----------------------------------------------------------------
C     UNDERFLOW TEST
C-----------------------------------------------------------------
      IF (RCZ.LT.(-ELIM)) GO TO 60
      IF (RCZ.GT.(-ALIM)) GO TO 100
      RCZ = RCZ + ALOG(APHI)
      IF (IFORM.EQ.2) RCZ = RCZ - 0.25E0*ALOG(AARG) - AIC
      IF (RCZ.GT.(-ELIM)) GO TO 80
   60 CONTINUE
      DO 70 I=1,NN
        Y(I) = CZERO
   70 CONTINUE
      NUF = NN
      RETURN
   80 CONTINUE
      ASCLE = 1.0E+3*R1MACH(1)/TOL
      CZ = CZ + CLOG(PHI)
      IF (IFORM.EQ.1) GO TO 90
      CZ = CZ - CMPLX(0.25E0,0.0E0)*CLOG(ARG) - CMPLX(AIC,0.0E0)
   90 CONTINUE
      AX = EXP(RCZ)/TOL
      AY = AIMAG(CZ)
      CZ = CMPLX(AX,0.0E0)*CMPLX(COS(AY),SIN(AY))
      CALL CUCHK(CZ, NW, ASCLE, TOL)
      IF (NW.EQ.1) GO TO 60
  100 CONTINUE
      IF (IKFLG.EQ.2) RETURN
```

```
      IF (N.EQ.1) RETURN
C-----------------------------------------------------------------------
C     SET UNDERFLOWS ON I SEQUENCE
C-----------------------------------------------------------------------
  110 CONTINUE
      GNU = FNU + FLOAT(NN-1)
      IF (IFORM.EQ.2) GO TO 120
      INIT = 0
      CALL CUNIK(ZR, GNU, IKFLG, 1, TOL, INIT, PHI, ZETA1, ZETA2, SUM,
     * CWRK)
      CZ = -ZETA1 + ZETA2
      GO TO 130
  120 CONTINUE
      CALL CUNHJ(ZN, GNU, 1, TOL, PHI, ARG, ZETA1, ZETA2, ASUM, BSUM)
      CZ = -ZETA1 + ZETA2
      AARG = CABS(ARG)
  130 CONTINUE
      IF (KODE.EQ.2) CZ = CZ - ZB
      APHI = CABS(PHI)
      RCZ = REAL(CZ)
      IF (RCZ.LT.(-ELIM)) GO TO 140
      IF (RCZ.GT.(-ALIM)) RETURN
      RCZ = RCZ + ALOG(APHI)
      IF (IFORM.EQ.2) RCZ = RCZ - 0.25E0*ALOG(AARG) - AIC
      IF (RCZ.GT.(-ELIM)) GO TO 150
  140 CONTINUE
      Y(NN) = CZERO
      NN = NN - 1
      NUF = NUF + 1
      IF (NN.EQ.0) RETURN
      GO TO 110
  150 CONTINUE
      ASCLE = 1.0E+3*R1MACH(1)/TOL
      CZ = CZ + CLOG(PHI)
      IF (IFORM.EQ.1) GO TO 160
      CZ = CZ - CMPLX(0.25E0,0.0E0)*CLOG(ARG) - CMPLX(AIC,0.0E0)
  160 CONTINUE
      AX = EXP(RCZ)/TOL
      AY = AIMAG(CZ)
      CZ = CMPLX(AX,0.0E0)*CMPLX(COS(AY),SIN(AY))
      CALL CUCHK(CZ, NW, ASCLE, TOL)
      IF (NW.EQ.1) GO TO 140
      RETURN
  170 CONTINUE
      NUF = -1
      RETURN
      END
      SUBROUTINE CWRSK(ZR, FNU, KODE, N, Y, NZ, CW, TOL, ELIM, ALIM)
C***BEGIN PROLOGUE  CWRSK
C***REFER TO  CBESI,CBESK
C
C     CWRSK COMPUTES THE I BESSEL FUNCTION FOR RE(Z).GE.0.0 BY
C     NORMALIZING THE I FUNCTION RATIOS FROM CRATI BY THE WRONSKIAN
C
C***ROUTINES CALLED  CBKNU,CRATI,R1MACH
C***END PROLOGUE  CWRSK
      COMPLEX CINU, CSCL, CT, CW, C1, C2, RCT, ST, Y, ZR
      REAL ACT, ACW, ALIM, ASCLE, ELIM, FNU, S1, S2, TOL, YY
      INTEGER I, KODE, N, NW, NZ
      DIMENSION Y(N), CW(2)
C-----------------------------------------------------------------------
```

```
C     I(FNU+I-1,Z) BY BACKWARD RECURRENCE FOR RATIOS
C     Y(I)=I(FNU+I,Z)/I(FNU+I-1,Z) FROM CRATI NORMALIZED BY THE
C     WRONSKIAN WITH K(FNU,Z) AND K(FNU+1,Z) FROM CBKNU.
C-----------------------------------------------------------------------
      NZ = 0
      CALL CBKNU(ZR, FNU, KODE, 2, CW, NW, TOL, ELIM, ALIM)
      IF (NW.NE.0) GO TO 50
      CALL CRATI(ZR, FNU, N, Y, TOL)
C-----------------------------------------------------------------------
C     RECUR FORWARD ON I(FNU+1,Z) = R(FNU,Z)*I(FNU,Z),
C     R(FNU+J-1,Z)=Y(J),  J=1,...,N
C-----------------------------------------------------------------------
      CINU = CMPLX(1.0E0,0.0E0)
      IF (KODE.EQ.1) GO TO 10
      YY = AIMAG(ZR)
      S1 = COS(YY)
      S2 = SIN(YY)
      CINU = CMPLX(S1,S2)
   10 CONTINUE
C-----------------------------------------------------------------------
C     ON LOW EXPONENT MACHINES THE K FUNCTIONS CAN BE CLOSE TO BOTH
C     THE UNDER AND OVERFLOW LIMITS AND THE NORMALIZATION MUST BE
C     SCALED TO PREVENT OVER OR UNDERFLOW. CUOIK HAS DETERMINED THAT
C     THE RESULT IS ON SCALE.
C-----------------------------------------------------------------------
      ACW = CABS(CW(2))
      ASCLE = 1.0E+3*R1MACH(1)/TOL
      CSCL = CMPLX(1.0E0,0.0E0)
      IF (ACW.GT.ASCLE) GO TO 20
      CSCL = CMPLX(1.0E0/TOL,0.0E0)
      GO TO 30
   20 CONTINUE
      ASCLE = 1.0E0/ASCLE
      IF (ACW.LT.ASCLE) GO TO 30
      CSCL = CMPLX(TOL,0.0E0)
   30 CONTINUE
      C1 = CW(1)*CSCL
      C2 = CW(2)*CSCL
      ST = Y(1)
C-----------------------------------------------------------------------
C     CINU=CINU*(CONJG(CT)/CABS(CT))*(1.0E0/CABS(CT) PREVENTS
C     UNDER- OR OVERFLOW PREMATURELY BY SQUARING CABS(CT)
C-----------------------------------------------------------------------
      CT = ZR*(C2+ST*C1)
      ACT = CABS(CT)
      RCT = CMPLX(1.0E0/ACT,0.0E0)
      CT = CONJG(CT)*RCT
      CINU = CINU*RCT*CT
      Y(1) = CINU*CSCL
      IF (N.EQ.1) RETURN
      DO 40 I=2,N
        CINU = ST*CINU
        ST = Y(I)
        Y(I) = CINU*CSCL
   40 CONTINUE
      RETURN
   50 CONTINUE
      NZ = -1
      IF(NW.EQ.(-2)) NZ=-2
      RETURN
      END
```

```
      FUNCTION GAMLN(Z,IERR)
C***BEGIN PROLOGUE  GAMLN
C***DATE WRITTEN   830501   (YYMMDD)
C***REVISION DATE  830501   (YYMMDD)
C***CATEGORY NO.  B5F
C***KEYWORDS  GAMMA FUNCTION,LOGARITHM OF GAMMA FUNCTION
C***AUTHOR  AMOS, DONALD E., SANDIA NATIONAL LABORATORIES
C***PURPOSE  TO COMPUTE THE LOGARITHM OF THE GAMMA FUNCTION
C***DESCRIPTION
C
C         GAMLN COMPUTES THE NATURAL LOG OF THE GAMMA FUNCTION FOR
C         Z.GT.0.  THE ASYMPTOTIC EXPANSION IS USED TO GENERATE VALUES
C         GREATER THAN ZMIN WHICH ARE ADJUSTED BY THE RECURSION
C         G(Z+1)=Z*G(Z) FOR Z.LE.ZMIN.  THE FUNCTION WAS MADE AS
C         PORTABLE AS POSSIBLE BY COMPUTIMG ZMIN FROM THE NUMBER OF BASE
C         10 DIGITS IN A WORD, RLN=AMAX1(-ALOG10(R1MACH(4)),0.5E-18)
C         LIMITED TO 18 DIGITS OF (RELATIVE) ACCURACY.
C
C         SINCE INTEGER ARGUMENTS ARE COMMON, A TABLE LOOK UP ON 100
C         VALUES IS USED FOR SPEED OF EXECUTION.
C
C     DESCRIPTION OF ARGUMENTS
C
C         INPUT
C           Z      - REAL ARGUMENT, Z.GT.0.0E0
C
C         OUTPUT
C           GAMLN  - NATURAL LOG OF THE GAMMA FUNCTION AT Z
C           IERR   - ERROR FLAG
C                    IERR=0, NORMAL RETURN, COMPUTATION COMPLETED
C                    IERR=1, Z.LE.0.0E0,    NO COMPUTATION
C
C***REFERENCES  COMPUTATION OF BESSEL FUNCTIONS OF COMPLEX
ARGUMENT
C                BY D. E. AMOS, SAND83-0083, MAY, 1983.
C***ROUTINES CALLED  I1MACH,R1MACH
C***END PROLOGUE  GAMLN
C
      INTEGER I, I1M, K, MZ, NZ, IERR, I1MACH
      REAL CF, CON, FLN, FZ, GLN, RLN, S, TLG, TRM, TST, T1, WDTOL, Z,
     * ZDMY, ZINC, ZM, ZMIN, ZP, ZSQ
      REAL R1MACH
      DIMENSION CF(22), GLN(100)
C           LNGAMMA(N), N=1,100
      DATA GLN(1), GLN(2), GLN(3), GLN(4), GLN(5), GLN(6), GLN(7),
     1     GLN(8), GLN(9), GLN(10), GLN(11), GLN(12), GLN(13), GLN(14),
     2     GLN(15), GLN(16), GLN(17), GLN(18), GLN(19), GLN(20),
     3     GLN(21), GLN(22)/
     4     0.00000000000000000E+00,     0.00000000000000000E+00,
     5     6.93147180559945309E-01,     1.79175946922805500E+00,
     6     3.17805383034794562E+00,     4.78749174278204599E+00,
     7     6.57925121201010100E+00,     8.52516136106541430E+00,
     8     1.06046029027452502E+01,     1.28018274800814696E+01,
     9     1.51044125730755153E+01,     1.75023078458738858E+01,
     A     1.99872144956618861E+01,     2.25521638531234229E+01,
     B     2.51912211827386815E+01,     2.78992713838408916E+01,
     C     3.06718601060806728E+01,     3.35050734501368889E+01,
     D     3.63954452080330536E+01,     3.93398841871994940E+01,
     E     4.23356164607534850E+01,     4.53801388984769080E+01/
      DATA GLN(23), GLN(24), GLN(25), GLN(26), GLN(27), GLN(28),
     1     GLN(29), GLN(30), GLN(31), GLN(32), GLN(33), GLN(34),
```

```
     2    GLN(35), GLN(36), GLN(37), GLN(38), GLN(39), GLN(40),
     3    GLN(41), GLN(42), GLN(43), GLN(44)/
     4    4.84711813518352239E+01,    5.16066755677643736E+01,
     5    5.47847293981123192E+01,    5.80036052229805199E+01,
     6    6.12617017610020020E+01,    6.45575386270063311E+01,
     7    6.78897431371815350E+01,    7.12570389671680090E+01,
     8    7.46582363488301644E+01,    7.80922235533153106E+01,
     9    8.15579594561150372E+01,    8.50544670175815174E+01,
     A    8.85808275421976788E+01,    9.21361756036870925E+01,
     B    9.57196945421432025E+01,    9.93306124547874269E+01,
     C    1.02968198614513813E+02,    1.06631760260643459E+02,
     D    1.10320639714757395E+02,    1.14034211781461703E+02,
     E    1.17771881399745072E+02,    1.21533081515438634E+02/
      DATA GLN(45), GLN(46), GLN(47), GLN(48), GLN(49), GLN(50),
     1    GLN(51), GLN(52), GLN(53), GLN(54), GLN(55), GLN(56),
     2    GLN(57), GLN(58), GLN(59), GLN(60), GLN(61), GLN(62),
     3    GLN(63), GLN(64), GLN(65), GLN(66)/
     4    1.25317271149356895E+02,    1.29123933639127215E+02,
     5    1.32952575035616310E+02,    1.36802722637326368E+02,
     6    1.40673923648234259E+02,    1.44565743946344886E+02,
     7    1.48477766951773032E+02,    1.52409592584497358E+02,
     8    1.56360836303078785E+02,    1.60331128216630907E+02,
     9    1.64320112263195181E+02,    1.68327445448427652E+02,
     A    1.72352797139162802E+02,    1.76395848406997352E+02,
     B    1.80456291417543771E+02,    1.84533828861449491E+02,
     C    1.88628173423671591E+02,    1.92739047287844902E+02,
     D    1.96866181672889994E+02,    2.01009316399281527E+02,
     E    2.05168199482641199E+02,    2.09342586752536836E+02/
      DATA GLN(67), GLN(68), GLN(69), GLN(70), GLN(71), GLN(72),
     1    GLN(73), GLN(74), GLN(75), GLN(76), GLN(77), GLN(78),
     2    GLN(79), GLN(80), GLN(81), GLN(82), GLN(83), GLN(84),
     3    GLN(85), GLN(86), GLN(87), GLN(88)/
     4    2.13532241494563261E+02,    2.17736934113954227E+02,
     5    2.21956441819130334E+02,    2.26190548323727593E+02,
     6    2.30439043565776952E+02,    2.34701723442818268E+02,
     7    2.38978389561834323E+02,    2.43268849002982714E+02,
     8    2.47572914096186884E+02,    2.51890402209723194E+02,
     9    2.56221135550009525E+02,    2.60564940971863209E+02,
     A    2.64921649798552801E+02,    2.69291097651019823E+02,
     B    2.73673124285693704E+02,    2.78067573440366143E+02,
     C    2.82474292687630396E+02,    2.86893133295426994E+02,
     D    2.91323950094270308E+02,    2.95766601350760624E+02,
     E    3.00220948647014132E+02,    3.04686856765668715E+02/
      DATA GLN(89), GLN(90), GLN(91), GLN(92), GLN(93), GLN(94),
     1    GLN(95), GLN(96), GLN(97), GLN(98), GLN(99), GLN(100)/
     2    3.09164193580146922E+02,    3.13652829949879062E+02,
     3    3.18152639620209327E+02,    3.22663499126726177E+02,
     4    3.27185287703775217E+02,    3.31717887196928473E+02,
     5    3.36261181979198477E+02,    3.40815058870799018E+02,
     6    3.45379407062266854E+02,    3.49954118040770237E+02,
     7    3.54539085519440809E+02,    3.59134205369575399E+02/
C           COEFFICIENTS OF ASYMPTOTIC EXPANSION
      DATA CF(1), CF(2), CF(3), CF(4), CF(5), CF(6), CF(7), CF(8),
     1    CF(9), CF(10), CF(11), CF(12), CF(13), CF(14), CF(15),
     2    CF(16), CF(17), CF(18), CF(19), CF(20), CF(21), CF(22)/
     3    8.33333333333333333E-02,   -2.77777777777777778E-03,
     4    7.93650793650793651E-04,   -5.95238095238095238E-04,
     5    8.41750841750841751E-04,   -1.91752691752691753E-03,
     6    6.41025641025641026E-03,   -2.95506535947712418E-02,
     7    1.79644372368830573E-01,   -1.39243221690590112E+00,
     8    1.34028640441683920E+01,   -1.56848284626002017E+02,
```

```
     9    2.19310333333333333E+03,   -3.61087712537249894E+04,
     A    6.91472268851313067E+05,   -1.52382215394074162E+07,
     B    3.82900751391414141E+08,   -1.08822660357843911E+10,
     C    3.47320283765002252E+11,   -1.23696021422692745E+13,
     D    4.88788064793079335E+14,   -2.13203339609193739E+16/
C
C          LN(2*PI)
      DATA CON                 /     1.83787706640934548E+00/
C
C***FIRST EXECUTABLE STATEMENT  GAMLN
      IERR=0
      IF (Z.LE.0.0E0) GO TO 70
      IF (Z.GT.101.0E0) GO TO 10
      NZ = INT(Z)
      FZ = Z - FLOAT(NZ)
      IF (FZ.GT.0.0E0) GO TO 10
      IF (NZ.GT.100) GO TO 10
      GAMLN = GLN(NZ)
      RETURN
   10 CONTINUE
      WDTOL = R1MACH(4)
      WDTOL = AMAX1(WDTOL,0.5E-18)
      I1M = I1MACH(11)
      RLN = R1MACH(5)*FLOAT(I1M)
      FLN = AMIN1(RLN,20.0E0)
      FLN = AMAX1(FLN,3.0E0)
      FLN = FLN - 3.0E0
      ZM = 1.8000E0 + 0.3875E0*FLN
      MZ = INT(ZM) + 1
      ZMIN = FLOAT(MZ)
      ZDMY = Z
      ZINC = 0.0E0
      IF (Z.GE.ZMIN) GO TO 20
      ZINC = ZMIN - FLOAT(NZ)
      ZDMY = Z + ZINC
   20 CONTINUE
      ZP = 1.0E0/ZDMY
      T1 = CF(1)*ZP
      S = T1
      IF (ZP.LT.WDTOL) GO TO 40
      ZSQ = ZP*ZP
      TST = T1*WDTOL
      DO 30 K=2,22
        ZP = ZP*ZSQ
        TRM = CF(K)*ZP
        IF (ABS(TRM).LT.TST) GO TO 40
        S = S + TRM
   30 CONTINUE
   40 CONTINUE
      IF (ZINC.NE.0.0E0) GO TO 50
      TLG = ALOG(Z)
      GAMLN = Z*(TLG-1.0E0) + 0.5E0*(CON-TLG) + S
      RETURN
   50 CONTINUE
      ZP = 1.0E0
      NZ = INT(ZINC)
      DO 60 I=1,NZ
        ZP = ZP*(Z+FLOAT(I-1))
   60 CONTINUE
      TLG = ALOG(ZDMY)
      GAMLN = ZDMY*(TLG-1.0E0) - ALOG(ZP) + 0.5E0*(CON-TLG) + S
```

```
      RETURN
C
C
   70 CONTINUE
      IERR=1
      RETURN
      END
*DECK I1MACH
      INTEGER FUNCTION I1MACH(I)
C***BEGIN PROLOGUE  I1MACH
C***DATE WRITTEN   750101   (YYMMDD)
C***REVISION DATE  890213   (YYMMDD)
C***CATEGORY NO.  R1
C***KEYWORDS  LIBRARY=SLATEC,TYPE=INTEGER(I1MACH-I),MACHINE
CONSTANTS
C***AUTHOR  FOX, P. A., (BELL LABS)
C           HALL, A. D., (BELL LABS)
C           SCHRYER, N. L., (BELL LABS)
C***PURPOSE  Returns integer machine dependent constants
C***DESCRIPTION
C
C     I1MACH can be used to obtain machine-dependent parameters
C     for the local machine environment.  It is a function
C     subroutine with one (input) argument, and can be called
C     as follows, for example
C
C          K = I1MACH(I)
C
C     where I=1,...,16.  The (output) value of K above is
C     determined by the (input) value of I.  The results for
C     various values of I are discussed below.
C
C  I/O unit numbers.
C    I1MACH( 1) = the standard input unit.
C    I1MACH( 2) = the standard output unit.
C    I1MACH( 3) = the standard punch unit.
C    I1MACH( 4) = the standard error message unit.
C
C  Words.
C    I1MACH( 5) = the number of bits per integer storage unit.
C    I1MACH( 6) = the number of characters per integer storage unit.
C
C  Integers.
C    assume integers are represented in the S-digit, base-A form
C
C               sign ( X(S-1)*A**(S-1) + ... + X(1)*A + X(0) )
C
C               where 0 .LE. X(I) .LT. A for I=0,...,S-1.
C    I1MACH( 7) = A, the base.
C    I1MACH( 8) = S, the number of base-A digits.
C    I1MACH( 9) = A**S - 1, the largest magnitude.
C
C  Floating-Point Numbers.
C    Assume floating-point numbers are represented in the T-digit,
C    base-B form
C               sign (B**E)*( (X(1)/B) + ... + (X(T)/B**T) )
C
C               where 0 .LE. X(I) .LT. B for I=1,...,T,
C               0 .LT. X(1), and EMIN .LE. E .LE. EMAX.
C    I1MACH(10) = B, the base.
C
```

```
C  Single-Precision
C    I1MACH(11) = T, the number of base-B digits.
C    I1MACH(12) = EMIN, the smallest exponent E.
C    I1MACH(13) = EMAX, the largest exponent E.
C
C  Double-Precision
C    I1MACH(14) = T, the number of base-B digits.
C    I1MACH(15) = EMIN, the smallest exponent E.
C    I1MACH(16) = EMAX, the largest exponent E.
C
C  To alter this function for a particular environment,
C  the desired set of DATA statements should be activated by
C  removing the C from column 1.  Also, the values of
C  I1MACH(1) - I1MACH(4) should be checked for consistency
C  with the local operating system.
C
C***REFERENCES  FOX P.A., HALL A.D., SCHRYER N.L.,*FRAMEWORK FOR A
C                 PORTABLE LIBRARY*, ACM TRANSACTIONS ON MATHEMATICAL
C                 SOFTWARE, VOL. 4, NO. 2, JUNE 1978, PP. 177-188.
C***ROUTINES CALLED  (NONE)
C***END PROLOGUE  I1MACH
C
      INTEGER IMACH(16),OUTPUT
      SAVE IMACH
      EQUIVALENCE (IMACH(4),OUTPUT)
C
C     MACHINE CONSTANTS FOR THE IBM PC
C
      DATA IMACH( 1) /    5 /
      DATA IMACH( 2) /    6 /
      DATA IMACH( 3) /    0 /
      DATA IMACH( 4) /    0 /
      DATA IMACH( 5) /   32 /
      DATA IMACH( 6) /    4 /
      DATA IMACH( 7) /    2 /
      DATA IMACH( 8) /   31 /
      DATA IMACH( 9) / 2147483647 /
      DATA IMACH(10) /    2 /
      DATA IMACH(11) /   24 /
      DATA IMACH(12) /  -125 /
      DATA IMACH(13) /   127 /
      DATA IMACH(14) /   53 /
      DATA IMACH(15) / -1021 /
      DATA IMACH(16) /  1023 /
C
C***FIRST EXECUTABLE STATEMENT  I1MACH
      IF (I .LT. 1  .OR.  I .GT. 16) GO TO 10
      I1MACH = IMACH(I)
      RETURN
   10 CONTINUE
      WRITE (UNIT = OUTPUT, FMT = 9000)
 9000 FORMAT ('1ERROR    1 IN I1MACH - I OUT OF BOUNDS')
C
C     CALL FDUMP
C
      STOP
      END
*DECK R1MACH
      REAL FUNCTION R1MACH(I)
C***BEGIN PROLOGUE  R1MACH
C***DATE WRITTEN   790101   (YYMMDD)
```

```
C***REVISION DATE  890213   (YYMMDD)
C***CATEGORY NO.  R1
C***KEYWORDS  LIBRARY=SLATEC,TYPE=SINGLE PRECISION(R1MACH-S D1MACH-
D),
C           MACHINE CONSTANTS
C***AUTHOR  FOX, P. A., (BELL LABS)
C         HALL, A. D., (BELL LABS)
C         SCHRYER, N. L., (BELL LABS)
C***PURPOSE  Returns single precision machine dependent constants
C***DESCRIPTION
C
C   R1MACH can be used to obtain machine-dependent parameters
C   for the local machine environment.  It is a function
C   subroutine with one (input) argument, and can be called
C   as follows, for example
C
C       A = R1MACH(I)
C
C   where I=1,...,5.  The (output) value of A above is
C   determined by the (input) value of I.  The results for
C   various values of I are discussed below.
C
C   Single-Precision Machine Constants
C   R1MACH(1) = B**(EMIN-1), the smallest positive magnitude.
C   R1MACH(2) = B**EMAX*(1 - B**(-T)), the largest magnitude.
C   R1MACH(3) = B**(-T), the smallest relative spacing.
C   R1MACH(4) = B**(1-T), the largest relative spacing.
C   R1MACH(5) = LOG10(B)
C
C   Assume single precision numbers are represented in the T-digit,
C   base-B form
C
C            sign (B**E)*( (X(1)/B) + ... + (X(T)/B**T) )
C
C   where 0 .LE. X(I) .LT. B for I=1,...,T, 0 .LT. X(1), and
C   EMIN .LE. E .LE. EMAX.
C
C   The values of B, T, EMIN and EMAX are provided in I1MACH as
C   follows:
C   I1MACH(10) = B, the base.
C   I1MACH(11) = T, the number of base-B digits.
C   I1MACH(12) = EMIN, the smallest exponent E.
C   I1MACH(13) = EMAX, the largest exponent E.
C
C   To alter this function for a particular environment,
C   the desired set of DATA statements should be activated by
C   removing the C from column 1.  Also, the values of
C   R1MACH(1) - R1MACH(4) should be checked for consistency
C   with the local operating system.
C
C***REFERENCES  FOX, P.A., HALL, A.D., SCHRYER, N.L, *FRAMEWORK FOR
C              A PORTABLE LIBRARY*, ACM TRANSACTIONS ON MATHE-
C              MATICAL SOFTWARE, VOL. 4, NO. 2, JUNE 1978,
C              PP. 177-188.
C***ROUTINES CALLED  XERROR
C***END PROLOGUE  R1MACH
      INTEGER SMALL(2)
      INTEGER LARGE(2)
      INTEGER RIGHT(2)
      INTEGER DIVER(2)
      INTEGER LOG10(2)
```

```
      REAL RMACH(5)
      SAVE RMACH
      EQUIVALENCE (RMACH(1),SMALL(1))
      EQUIVALENCE (RMACH(2),LARGE(1))
      EQUIVALENCE (RMACH(3),RIGHT(1))
      EQUIVALENCE (RMACH(4),DIVER(1))
      EQUIVALENCE (RMACH(5),LOG10(1))
C
C     MACHINE CONSTANTS FOR THE IBM PC
C
      DATA SMALL(1) /    8420761 /
      DATA LARGE(1) / 2139081118 /
      DATA RIGHT(1) /  863997169 /
      DATA DIVER(1) /  872385777 /
      DATA LOG10(1) / 1050288283 /
C
C***FIRST EXECUTABLE STATEMENT  R1MACH
      IF (I .LT. 1  .OR.  I .GT. 5)
     1   CALL XERROR ('R1MACH -- I OUT OF BOUNDS', 25, 1, 2)
      R1MACH = RMACH(I)
      RETURN
      END
      SUBROUTINE XERROR(MESS,NMESS,L1,L2)
C
C     THIS IS A DUMMY XERROR ROUTINE TO PRINT ERROR MESSAGES WITH
NMESS
C     CHARACTERS. L1 AND L2 ARE DUMMY PARAMETERS TO MAKE THIS CALL
C     COMPATIBLE WITH THE SLATEC XERROR ROUTINE. THIS IS A FORTRAN 77
C     ROUTINE.
C
      CHARACTER*(*) MESS
      NN=NMESS/70
      NR=NMESS-70*NN
      IF(NR.NE.0) NN=NN+1
      K=1
      PRINT 900
  900 FORMAT(/)
      DO 10 I=1,NN
        KMIN=MIN0(K+69,NMESS)
        PRINT *, MESS(K:KMIN)
        K=K+70
   10 CONTINUE
      PRINT 900
      RETURN
      END


* ==================================================================
* NIST Guide to Available Math Software.
* Fullsource for module BESJ0 from package CMLIB.
* Retrieved from CAMSUN on Tue Jan  6 07:57:07 1998.
* ==================================================================
      FUNCTION BESJ0(X)
C***BEGIN PROLOGUE  BESJ0
C***DATE WRITTEN   770401   (YYMMDD)
C***REVISION DATE  820801   (YYMMDD)
C***CATEGORY NO.  C10A1
C***KEYWORDS  BESSEL FUNCTION,FIRST KIND,ORDER ZERO,SPECIAL FUNCTION
C***AUTHOR  FULLERTON, W., (LANL)
C***PURPOSE  Computes the Bessel function of the first kind of order
C            zero
```

```
C***DESCRIPTION
C
C BESJ0(X) calculates the Bessel function of the first kind of
C order zero for real argument X.
C
C Series for BJ0       on the interval  0.         to  1.60000D+01
C                              with weighted error   7.47E-18
C                               log weighted error  17.13
C                         significant figures required  16.98
C                             decimal places required  17.68
C
C Series for BM0       on the interval  0.         to  6.25000D-02
C                              with weighted error   4.98E-17
C                               log weighted error  16.30
C                         significant figures required  14.97
C                             decimal places required  16.96
C
C Series for BTH0      on the interval  0.         to  6.25000D-02
C                              with weighted error   3.67E-17
C                               log weighted error  16.44
C                         significant figures required  15.53
C                             decimal places required  17.13
C***REFERENCES  (NONE)
C***ROUTINES CALLED  CSEVL,INITS,R1MACH,XERROR
C***END PROLOGUE  BESJ0
      DIMENSION BJ0CS(13), BM0CS(21), BTH0CS(24)
      DATA BJ0 CS( 1) /    .1002541619 68939137E0 /
      DATA BJ0 CS( 2) /   -.6652230077 64405132E0 /
      DATA BJ0 CS( 3) /    .2489837034 98281314E0 /
      DATA BJ0 CS( 4) /   -.0332527231 700357697E0 /
      DATA BJ0 CS( 5) /    .0023114179 304694015E0 /
      DATA BJ0 CS( 6) /   -.0000991127 741995080E0 /
      DATA BJ0 CS( 7) /    .0000028916 708643998E0 /
      DATA BJ0 CS( 8) /   -.0000000612 108586630E0 /
      DATA BJ0 CS( 9) /    .0000000009 838650793E0 /
      DATA BJ0 CS(10) /   -.0000000000 124235515E0 /
      DATA BJ0 CS(11) /    .0000000000 001265433E0 /
      DATA BJ0 CS(12) /   -.0000000000 000010619E0 /
      DATA BJ0 CS(13) /    .0000000000 000000074E0 /
      DATA BM0 CS( 1) /    .0928496163 7381644E0 /
      DATA BM0 CS( 2) /   -.0014298770 7403484E0 /
      DATA BM0 CS( 3) /    .0000283057 9271257E0 /
      DATA BM0 CS( 4) /   -.0000014330 0611424E0 /
      DATA BM0 CS( 5) /    .0000001202 8628046E0 /
      DATA BM0 CS( 6) /   -.0000000139 7113013E0 /
      DATA BM0 CS( 7) /    .0000000020 4076188E0 /
      DATA BM0 CS( 8) /   -.0000000003 5399669E0 /
      DATA BM0 CS( 9) /    .0000000000 7024759E0 /
      DATA BM0 CS(10) /   -.0000000000 1554107E0 /
      DATA BM0 CS(11) /    .0000000000 0376226E0 /
      DATA BM0 CS(12) /   -.0000000000 0098282E0 /
      DATA BM0 CS(13) /    .0000000000 0027408E0 /
      DATA BM0 CS(14) /   -.0000000000 0008091E0 /
      DATA BM0 CS(15) /    .0000000000 0002511E0 /
      DATA BM0 CS(16) /   -.0000000000 0000814E0 /
      DATA BM0 CS(17) /    .0000000000 0000275E0 /
      DATA BM0 CS(18) /   -.0000000000 0000096E0 /
      DATA BM0 CS(19) /    .0000000000 0000034E0 /
      DATA BM0 CS(20) /   -.0000000000 0000012E0 /
      DATA BM0 CS(21) /    .0000000000 0000004E0 /
      DATA BTH0CS( 1) /   -.2463916377 4300119E0 /
```

```
      DATA BTH0CS( 2) /    .0017370983 07508963E0 /
      DATA BTH0CS( 3) /   -.0000621836 33402968E0 /
      DATA BTH0CS( 4) /    .0000043680 50165742E0 /
      DATA BTH0CS( 5) /   -.0000004560 93019869E0 /
      DATA BTH0CS( 6) /    .0000000621 97400101E0 /
      DATA BTH0CS( 7) /   -.0000000103 00442889E0 /
      DATA BTH0CS( 8) /    .0000000019 79526776E0 /
      DATA BTH0CS( 9) /   -.0000000004 28198396E0 /
      DATA BTH0CS(10) /    .0000000001 02035840E0 /
      DATA BTH0CS(11) /   -.0000000000 26363898E0 /
      DATA BTH0CS(12) /    .0000000000 07297935E0 /
      DATA BTH0CS(13) /   -.0000000000 02144188E0 /
      DATA BTH0CS(14) /    .0000000000 00663693E0 /
      DATA BTH0CS(15) /   -.0000000000 00215126E0 /
      DATA BTH0CS(16) /    .0000000000 00072659E0 /
      DATA BTH0CS(17) /   -.0000000000 00025465E0 /
      DATA BTH0CS(18) /    .0000000000 00009229E0 /
      DATA BTH0CS(19) /   -.0000000000 00003448E0 /
      DATA BTH0CS(20) /    .0000000000 00001325E0 /
      DATA BTH0CS(21) /   -.0000000000 00000522E0 /
      DATA BTH0CS(22) /    .0000000000 00000210E0 /
      DATA BTH0CS(23) /   -.0000000000 00000087E0 /
      DATA BTH0CS(24) /    .0000000000 00000036E0 /
      DATA PI4 / 0.7853981633 9744831E0 /
      DATA NTJ0, NTM0, NTTH0, XSML, XMAX / 3*0, 2*0./
C***FIRST EXECUTABLE STATEMENT  BESJ0
      IF (NTJ0.NE.0) GO TO 10
      NTJ0 = INITS (BJ0CS, 13, 0.1*R1MACH1(3))
      NTM0 = INITS (BM0CS, 21, 0.1*R1MACH1(3))
      NTTH0 = INITS (BTH0CS, 24, 0.1*R1MACH1(3))
      XSML = SQRT (4.0*R1MACH1(3))
      XMAX = 1.0/R1MACH1(4)
 10   Y = ABS(X)
      IF (Y.GT.4.0) GO TO 20
      BESJ0 = 1.0
      IF (Y.GT.XSML) BESJ0 = CSEVL (.125*Y*Y-1., BJ0CS, NTJ0)
      RETURN
 20   IF (Y.GT.XMAX) CALL XERROR1 (  'BESJ0   NO PRECISION BECAUSE ABS(X)
     1 IS BIG', 42, 1, 2)
      Z = 32.0/Y**2 - 1.0
      AMPL = (0.75 + CSEVL (Z, BM0CS, NTM0)) / SQRT(Y)
      THETA = Y - PI4 + CSEVL (Z, BTH0CS, NTTH0) / Y
      BESJ0 = AMPL * COS (THETA)
      RETURN
      END
      FUNCTION CSEVL(X,CS,N)
C***BEGIN PROLOGUE  CSEVL
C***DATE WRITTEN   770401   (YYMMDD)
C***REVISION DATE  820801   (YYMMDD)
C***CATEGORY NO.  C3A2
C***KEYWORDS  CHEBYSHEV,FNLIB,SPECIAL FUNCTION
C***AUTHOR  FULLERTON, W., (LANL)
C***PURPOSE  Evaluate the N-term Chebyshev series CS at X.
C***DESCRIPTION
C
C Evaluate the N-term Chebyshev series CS at X.  Adapted from
C R. Broucke, Algorithm 446, C.A.C.M., 16, 254 (1973). Also see Fox
C and Parker, Chebyshev Polynomials in Numerical Analysis, Oxford Press,
C page 56.
C
C      Input Arguments --
```

```
C X    value at which the series is to be evaluated.
C CS   array of N terms of a Chebyshev series.  In eval-
C      uating CS, only half the first coefficient is summed.
C N    number of terms in array CS.
C***REFERENCES  (NONE)
C***ROUTINES CALLED  XERROR
C***END PROLOGUE  CSEVL
C
      DIMENSION CS(1)
C***FIRST EXECUTABLE STATEMENT  CSEVL
      IF(N.LT.1) CALL XERROR1( 'CSEVL   NUMBER OF TERMS LE 0', 28, 2,2)
      IF(N.GT.1000) CALL XERROR1 ( 'CSEVL   NUMBER OF TERMS GT 1000',
     1   31,3,2)
      IF (X.LT. -1.0 .OR. X.GT. 1.0) CALL XERROR1( 'CSEVL   X OUTSIDE (-
     11,+1)', 25, 1, 1)
      B1=0.
      B0=0.
      TWOX=2.*X
      DO 10 I=1,N
      B2=B1
      B1=B0
      NI=N+1-I
      B0=TWOX*B1-B2+CS(NI)
 10    CONTINUE
      CSEVL = 0.5 * (B0-B2)
      RETURN
     END
     FUNCTION INITS(OS,NOS,ETA)
C***BEGIN PROLOGUE  INITS
C***DATE WRITTEN   770401   (YYMMDD)
C***REVISION DATE  820801   (YYMMDD)
C***CATEGORY NO.  C3A2
C***KEYWORDS  INITIALIZE,ORTHOGONAL SERIES,SPECIAL FUNCTION
C***AUTHOR  FULLERTON, W., (LANL)
C***PURPOSE  Initializes an orthogonal series so that it defines the
C            number of terms to carry in the series to meet a specified
C            error.
C***DESCRIPTION
C
C Initialize the orthogonal series so that INITS is the number of terms
C needed to insure the error is no larger than ETA.  Ordinarily, ETA
C will be chosen to be one-tenth machine precision.
C
C             Input Arguments --
C OS     array of NOS coefficients in an orthogonal series.
C NOS    number of coefficients in OS.
C ETA    requested accuracy of series.
C***REFERENCES  (NONE)
C***ROUTINES CALLED  XERROR
C***END PROLOGUE  INITS
     DIMENSION OS(NOS)
C***FIRST EXECUTABLE STATEMENT  INITS
     IF (NOS.LT.1) CALL XERROR1( 'INITS   NUMBER OF COEFFICIENTS LT 1',
    1 35, 2, 2)
     ERR = 0.
     DO 10 II=1,NOS
       I = NOS + 1 - II
       ERR = ERR + ABS(OS(I))
       IF (ERR.GT.ETA) GO TO 20
 10  CONTINUE
 20  IF (I.EQ.NOS) CALL XERROR1( 'INITS   ETA MAY BE TOO SMALL', 28,
```

```
     1  1, 2)
      INITS = I
      RETURN
      END
      REAL FUNCTION R1MACH1(I)
C***BEGIN PROLOGUE  R1MACH
C***DATE WRITTEN   790101   (YYMMDD)
C***REVISION DATE  910131   (YYMMDD)
C***CATEGORY NO.  R1
C***KEYWORDS  MACHINE CONSTANTS
C***AUTHOR  FOX, P. A., (BELL LABS)
C           HALL, A. D., (BELL LABS)
C           SCHRYER, N. L., (BELL LABS)
C***PURPOSE  Returns single precision machine dependent constants
C***DESCRIPTION
C
C     This is the CMLIB version of R1MACH, the real machine
C     constants subroutine originally developed for the PORT library.
C
C     R1MACH can be used to obtain machine-dependent parameters
C     for the local machine environment.  It is a function
C     subroutine with one (input) argument, and can be called
C     as follows, for example
C
C          A = R1MACH(I)
C
C     where I=1,...,5.  The (output) value of A above is
C     determined by the (input) value of I.  The results for
C     various values of I are discussed below.
C
C  Single-Precision Machine Constants
C  R1MACH(1) = B**(EMIN-1), the smallest positive magnitude.
C  R1MACH(2) = B**EMAX*(1 - B**(-T)), the largest magnitude.
C  R1MACH(3) = B**(-T), the smallest relative spacing.
C  R1MACH(4) = B**(1-T), the largest relative spacing.
C  R1MACH(5) = LOG10(B)
C***REFERENCES  FOX, P.A., HALL, A.D., SCHRYER, N.L, *FRAMEWORK FOR
C                 A PORTABLE LIBRARY*, ACM TRANSACTIONS ON MATHE-
C                 MATICAL SOFTWARE, VOL. 4, NO. 2, JUNE 1978,
C                 PP. 177-188.
C***ROUTINES CALLED  XERROR
C***END PROLOGUE  R1MACH
      INTEGER SMALL(2)
      INTEGER LARGE(2)
      INTEGER RIGHT(2)
      INTEGER DIVER(2)
      INTEGER LOG10(2)
      REAL RMACH(5)
      EQUIVALENCE (RMACH(1),SMALL(1))
      EQUIVALENCE (RMACH(2),LARGE(1))
      EQUIVALENCE (RMACH(3),RIGHT(1))
      EQUIVALENCE (RMACH(4),DIVER(1))
      EQUIVALENCE (RMACH(5),LOG10(1))
C
C     MACHINE CONSTANTS FOR IEEE ARITHMETIC MACHINES, SUCH AS THE AT&T
C     3B SERIES, MOTOROLA 68000 BASED MACHINES (E.G. SUN 3 AND AT&T
C     PC 7300), AND 8087 BASED MICROS (E.G. IBM PC AND AT&T 6300).
C
C === MACHINE = IEEE.MOST-SIG-BYTE-FIRST
C === MACHINE = IEEE.LEAST-SIG-BYTE-FIRST
C === MACHINE = SUN
```

```
C === MACHINE = 68000
C === MACHINE = 8087
C === MACHINE = IBM.PC
C === MACHINE = ATT.3B
C === MACHINE = ATT.6300
C === MACHINE = ATT.7300
      DATA SMALL(1) /    8388608 /
      DATA LARGE(1) /  2139095039 /
      DATA RIGHT(1) /   864026624 /
      DATA DIVER(1) /   872415232 /
      DATA LOG10(1) /  1050288283 /
C***FIRST EXECUTABLE STATEMENT  R1MACH
     IF (I .LT. 1  .OR.  I .GT. 5)
    1   CALL XERROR1( 'R1MACH -- I OUT OF BOUNDS',25,1,2)
     R1MACH1= RMACH(I)
     RETURN
     END
     SUBROUTINE XERROR1(MESSG,NMESSG,NERR,LEVEL)
C***BEGIN PROLOGUE  XERROR
C***DATE WRITTEN   790801   (YYMMDD)
C***REVISION DATE  820801   (YYMMDD)
C***CATEGORY NO.  R3C
C***KEYWORDS  ERROR,XERROR PACKAGE
C***AUTHOR  JONES, R. E., (SNLA)
C***PURPOSE  Processes an error (diagnostic) message.
C***DESCRIPTION
C     Abstract
C        XERROR processes a diagnostic message, in a manner
C        determined by the value of LEVEL and the current value
C        of the library error control flag, KONTRL.
C        (See subroutine XSETF for details.)
C
C     Description of Parameters
C      --Input--
C        MESSG - the Hollerith message to be processed, containing
C                no more than 72 characters.
C        NMESSG- the actual number of characters in MESSG.
C        NERR  - the error number associated with this message.
C                NERR must not be zero.
C        LEVEL - error category.
C                =2 means this is an unconditionally fatal error.
C                =1 means this is a recoverable error.  (I.e., it is
C                   non-fatal if XSETF has been appropriately called.)
C                =0 means this is a warning message only.
C                =-1 means this is a warning message which is to be
C                   printed at most once, regardless of how many
C                   times this call is executed.
C
C     Examples
C        CALL XERROR('SMOOTH -- NUM WAS ZERO.',23,1,2)
C        CALL XERROR('INTEG  -- LESS THAN FULL ACCURACY ACHIEVED.',
C                    43,2,1)
C        CALL XERROR('ROOTER -- ACTUAL ZERO OF F FOUND BEFORE INTERVAL F
C    1ULLY COLLAPSED.',65,3,0)
C        CALL XERROR('EXP    -- UNDERFLOWS BEING SET TO ZERO.',39,1,-1)
C
C     Latest revision ---  19 MAR 1980
C     Written by Ron Jones, with SLATEC Common Math Library Subcommittee
C***REFERENCES  JONES R.E., KAHANER D.K., "XERROR, THE SLATEC ERROR-
C                 HANDLING PACKAGE", SAND82-0800, SANDIA LABORATORIES,
C                 1982.
```

```
C***ROUTINES CALLED  XERRWV
C***END PROLOGUE  XERROR
      CHARACTER*(*) MESSG
C***FIRST EXECUTABLE STATEMENT  XERROR
      CALL XERRWV(MESSG,NMESSG,NERR,LEVEL,0,0,0,0,0.,0.)
      RETURN
      END
      SUBROUTINE XERRWV(MESSG,NMESSG,NERR,LEVEL,NI,I1,I2,NR,R1,R2)
C***BEGIN PROLOGUE  XERRWV
C***DATE WRITTEN   800319   (YYMMDD)
C***REVISION DATE  820801   (YYMMDD)
C***CATEGORY NO.  R3C
C***KEYWORDS  ERROR,XERROR PACKAGE
C***AUTHOR  JONES, R. E., (SNLA)
C***PURPOSE  Processes error message allowing 2 integer and two real
C            values to be included in the message.
C***DESCRIPTION
C     Abstract
C        XERRWV processes a diagnostic message, in a manner
C        determined by the value of LEVEL and the current value
C        of the library error control flag, KONTRL.
C        (See subroutine XSETF for details.)
C        In addition, up to two integer values and two real
C        values may be printed along with the message.
C
C     Description of Parameters
C      --Input--
C        MESSG - the Hollerith message to be processed.
C        NMESSG- the actual number of characters in MESSG.
C        NERR  - the error number associated with this message.
C                NERR must not be zero.
C        LEVEL - error category.
C                =2 means this is an unconditionally fatal error.
C                =1 means this is a recoverable error.  (I.e., it is
C                   non-fatal if XSETF has been appropriately called.)
C                =0 means this is a warning message only.
C                =-1 means this is a warning message which is to be
C                   printed at most once, regardless of how many
C                   times this call is executed.
C        NI    - number of integer values to be printed. (0 to 2)
C        I1    - first integer value.
C        I2    - second integer value.
C        NR    - number of real values to be printed. (0 to 2)
C        R1    - first real value.
C        R2    - second real value.
C
C     Examples
C        CALL XERRWV('SMOOTH -- NUM (=I1) WAS ZERO.',29,1,2,
C    1   1,NUM,0,0,0.,0.)
C        CALL XERRWV('QUADXY -- REQUESTED ERROR (R1) LESS THAN MINIMUM (
C    1R2).,54,77,1,0,0,0,2,ERRREQ,ERRMIN)
C
C     Latest revision ---  19 MAR 1980
C     Written by Ron Jones, with SLATEC Common Math Library Subcommittee
C***REFERENCES  JONES R.E., KAHANER D.K., "XERROR, THE SLATEC ERROR-
C                 HANDLING PACKAGE", SAND82-0800, SANDIA LABORATORIES,
C                 1982.
C***ROUTINES CALLED  FDUMP,I1MACH,J4SAVE,XERABT,XERCTL,XERPRT,XERSAV,
C                    XGETUA
C***END PROLOGUE  XERRWV
      CHARACTER*(*) MESSG
```

```
      CHARACTER*20 LFIRST
      CHARACTER*37 FORM
      DIMENSION LUN(5)
C     GET FLAGS
C***FIRST EXECUTABLE STATEMENT  XERRWV
      LKNTRL = J4SAVE(2,0,.FALSE.)
      MAXMES = J4SAVE(4,0,.FALSE.)
C     CHECK FOR VALID INPUT
      IF ((NMESSG.GT.0).AND.(NERR.NE.0).AND.
     1    (LEVEL.GE.(-1)).AND.(LEVEL.LE.2)) GO TO 10
         IF (LKNTRL.GT.0) CALL XERPRT('FATAL ERROR IN...',17)
         CALL XERPRT('XERROR -- INVALID INPUT',23)
         IF (LKNTRL.GT.0) CALL FDUMP
         IF (LKNTRL.GT.0) CALL XERPRT('JOB ABORT DUE TO FATAL ERROR.',
     1  29)
         IF (LKNTRL.GT.0) CALL XERSAV(' ',0,0,0,KDUMMY)
         CALL XERABT('XERROR -- INVALID INPUT',23)
         RETURN
   10 CONTINUE
C     RECORD MESSAGE
      JUNK = J4SAVE(1,NERR,.TRUE.)
      CALL XERSAV(MESSG,NMESSG,NERR,LEVEL,KOUNT)
C     LET USER OVERRIDE
      LFIRST = MESSG
      LMESSG = NMESSG
      LERR = NERR
      LLEVEL = LEVEL
      CALL XERCTL(LFIRST,LMESSG,LERR,LLEVEL,LKNTRL)
C     RESET TO ORIGINAL VALUES
      LMESSG = NMESSG
      LERR = NERR
      LLEVEL = LEVEL
      LKNTRL = MAX0(-2,MIN0(2,LKNTRL))
      MKNTRL = IABS(LKNTRL)
C     DECIDE WHETHER TO PRINT MESSAGE
      IF ((LLEVEL.LT.2).AND.(LKNTRL.EQ.0)) GO TO 100
      IF (((LLEVEL.EQ.(-1)).AND.(KOUNT.GT.MIN0(1,MAXMES)))
     1.OR.((LLEVEL.EQ.0)   .AND.(KOUNT.GT.MAXMES))
     2.OR.((LLEVEL.EQ.1)   .AND.(KOUNT.GT.MAXMES).AND.(MKNTRL.EQ.1))
     3.OR.((LLEVEL.EQ.2)   .AND.(KOUNT.GT.MAX0(1,MAXMES)))) GO TO 100
         IF (LKNTRL.LE.0) GO TO 20
            CALL XERPRT(' ',1)
C           INTRODUCTION
            IF (LLEVEL.EQ.(-1)) CALL XERPRT
     1('WARNING MESSAGE...THIS MESSAGE WILL ONLY BE PRINTED ONCE.',57)
            IF (LLEVEL.EQ.0) CALL XERPRT('WARNING IN...',13)
            IF (LLEVEL.EQ.1) CALL XERPRT
     1      ('RECOVERABLE ERROR IN...',23)
            IF (LLEVEL.EQ.2) CALL XERPRT('FATAL ERROR IN...',17)
   20    CONTINUE
C        MESSAGE
         CALL XERPRT(MESSG,LMESSG)
         CALL XGETUA(LUN,NUNIT)
         ISIZEI = LOG10(FLOAT(I1MACH1(9))) + 1.0
         ISIZEF = LOG10(FLOAT(I1MACH1(10))**I1MACH1(11)) + 1.0
         DO 50 KUNIT=1,NUNIT
            IUNIT = LUN(KUNIT)
            IF (IUNIT.EQ.0) IUNIT = I1MACH1(4)
            DO 22 I=1,MIN(NI,2)
               WRITE (FORM,21) I,ISIZEI
   21          FORMAT ('(11X,21HIN ABOVE MESSAGE, I',I1,'=,I',I2,')   ')
```

```
            IF (I.EQ.1) WRITE (IUNIT,FORM) I1
            IF (I.EQ.2) WRITE (IUNIT,FORM) I2
   22     CONTINUE
          DO 24 I=1,MIN(NR,2)
            WRITE (FORM,23) I,ISIZEF+10,ISIZEF
   23         FORMAT ('(11X,21HIN ABOVE MESSAGE, R',I1,'=,E',
     1        I2,'.',I2,')')
            IF (I.EQ.1) WRITE (IUNIT,FORM) R1
            IF (I.EQ.2) WRITE (IUNIT,FORM) R2
   24     CONTINUE
          IF (LKNTRL.LE.0) GO TO 40
C             ERROR NUMBER
            WRITE (IUNIT,30) LERR
   30         FORMAT (15H ERROR NUMBER =,I10)
   40     CONTINUE
   50   CONTINUE
C       TRACE-BACK
        IF (LKNTRL.GT.0) CALL FDUMP
  100 CONTINUE
      IFATAL = 0
      IF ((LLEVEL.EQ.2).OR.((LLEVEL.EQ.1).AND.(MKNTRL.EQ.2)))
     1IFATAL = 1
C     QUIT HERE IF MESSAGE IS NOT FATAL
      IF (IFATAL.LE.0) RETURN
      IF ((LKNTRL.LE.0).OR.(KOUNT.GT.MAX0(1,MAXMES))) GO TO 120
C        PRINT REASON FOR ABORT
         IF (LLEVEL.EQ.1) CALL XERPRT
     1   ('JOB ABORT DUE TO UNRECOVERED ERROR.',35)
         IF (LLEVEL.EQ.2) CALL XERPRT
     1   ('JOB ABORT DUE TO FATAL ERROR.',29)
C        PRINT ERROR SUMMARY
         CALL XERSAV(' ',-1,0,0,KDUMMY)
  120 CONTINUE
C     ABORT
      IF ((LLEVEL.EQ.2).AND.(KOUNT.GT.MAX0(1,MAXMES))) LMESSG = 0
      CALL XERABT(MESSG,LMESSG)
      RETURN
      END
      SUBROUTINE XERSAV(MESSG,NMESSG,NERR,LEVEL,ICOUNT)
C***BEGIN PROLOGUE  XERSAV
C***DATE WRITTEN   800319   (YYMMDD)
C***REVISION DATE  820801   (YYMMDD)
C***CATEGORY NO.  Z
C***KEYWORDS  ERROR,XERROR PACKAGE
C***AUTHOR  JONES, R. E., (SNLA)
C***PURPOSE  Records that an error occurred.
C***DESCRIPTION
C     Abstract
C        Record that this error occurred.
C
C     Description of Parameters
C     --Input--
C       MESSG, NMESSG, NERR, LEVEL are as in XERROR,
C       except that when NMESSG=0 the tables will be
C       dumped and cleared, and when NMESSG is less than zero the
C       tables will be dumped and not cleared.
C     --Output--
C       ICOUNT will be the number of times this message has
C       been seen, or zero if the table has overflowed and
C       does not contain this message specifically.
C       When NMESSG=0, ICOUNT will not be altered.
```

```
C
C    Written by Ron Jones, with SLATEC Common Math Library Subcommittee
C    Latest revision ---  19 Mar 1980
C***REFERENCES  JONES R.E., KAHANER D.K., "XERROR, THE SLATEC ERROR-
C                HANDLING PACKAGE", SAND82-0800, SANDIA LABORATORIES,
C                1982.
C***ROUTINES CALLED  I1MACH,S88FMT,XGETUA
C***END PROLOGUE  XERSAV
      INTEGER LUN(5)
      CHARACTER*(*) MESSG
      CHARACTER*20 MESTAB(10),MES
      DIMENSION NERTAB(10),LEVTAB(10),KOUNT(10)
      SAVE MESTAB,NERTAB,LEVTAB,KOUNT,KOUNTX
C    NEXT TWO DATA STATEMENTS ARE NECESSARY TO PROVIDE A BLANK
C    ERROR TABLE INITIALLY
      DATA KOUNT(1),KOUNT(2),KOUNT(3),KOUNT(4),KOUNT(5),
     1     KOUNT(6),KOUNT(7),KOUNT(8),KOUNT(9),KOUNT(10)
     2     /0,0,0,0,0,0,0,0,0,0/
      DATA KOUNTX/0/
C***FIRST EXECUTABLE STATEMENT  XERSAV
      IF (NMESSG.GT.0) GO TO 80
C    DUMP THE TABLE
         IF (KOUNT(1).EQ.0) RETURN
C       PRINT TO EACH UNIT
         CALL XGETUA(LUN,NUNIT)
         DO 60 KUNIT=1,NUNIT
            IUNIT = LUN(KUNIT)
            IF (IUNIT.EQ.0) IUNIT = I1MACH1(4)
C          PRINT TABLE HEADER
            WRITE (IUNIT,10)
   10       FORMAT (32H0          ERROR MESSAGE SUMMARY/
```

```
      IF (LEVEL.NE.LEVTAB(I)) GO TO 90
      GO TO 100
   90 CONTINUE
C     THREE POSSIBLE CASES...
C     TABLE IS FULL
      KOUNTX = KOUNTX+1
      ICOUNT = 1
      RETURN
C     MESSAGE FOUND IN TABLE
  100    KOUNT(II) = KOUNT(II) + 1
      ICOUNT = KOUNT(II)
      RETURN
C     EMPTY SLOT FOUND FOR NEW MESSAGE
  110    MESTAB(II) = MES
      NERTAB(II) = NERR
      LEVTAB(II) = LEVEL
      KOUNT(II)  = 1
      ICOUNT = 1
      RETURN
    END
    SUBROUTINE XGETUA(IUNITA,N)
C***BEGIN PROLOGUE  XGETUA
C***DATE WRITTEN   790801   (YYMMDD)
C***REVISION DATE  820801   (YYMMDD)
C***CATEGORY NO.  R3C
C***KEYWORDS  ERROR,XERROR PACKAGE
C***AUTHOR  JONES, R. E., (SNLA)
C***PURPOSE  Returns unit number(s) to which error messages are being
C            sent.
C***DESCRIPTION
C     Abstract
C        XGETUA may be called to determine the unit number or numbers
C        to which error messages are being sent.
C        These unit numbers may have been set by a call to XSETUN,
C        or a call to XSETUA, or may be a default value.
C
C     Description of Parameters
C      --Output--
C        IUNIT - an array of one to five unit numbers, depending
C                on the value of N.  A value of zero refers to the
C                default unit, as defined by the I1MACH machine
C                constant routine.  Only IUNIT(1),...,IUNIT(N) are
C                defined by XGETUA.  The values of IUNIT(N+1),...,
C                IUNIT(5) are not defined (for N .LT. 5) or altered
C                in any way by XGETUA.
C        N     - the number of units to which copies of the
C                error messages are being sent.  N will be in the
C                range from 1 to 5.
C
C     Latest revision ---  19 MAR 1980
C     Written by Ron Jones, with SLATEC Common Math Library Subcommittee
C***REFERENCES  JONES R.E., KAHANER D.K., "XERROR, THE SLATEC ERROR-
C                 HANDLING PACKAGE", SAND82-0800, SANDIA LABORATORIES,
C                 1982.
C***ROUTINES CALLED  J4SAVE
C***END PROLOGUE  XGETUA
    DIMENSION IUNITA(5)
C***FIRST EXECUTABLE STATEMENT  XGETUA
    N = J4SAVE(5,0,.FALSE.)
    DO 30 I=1,N
      INDEX = I+4
```

```
C
C    Written by Ron Jones, with SLATEC Common Math Library Subcommittee
C    Latest revision --- 19 Mar 1980
C***REFERENCES  JONES R.E., KAHANER D.K., "XERROR, THE SLATEC ERROR-
C                 HANDLING PACKAGE", SAND82-0800, SANDIA LABORATORIES,
C                 1982.
C***ROUTINES CALLED  I1MACH,S88FMT,XGETUA
C***END PROLOGUE  XERSAV
      INTEGER LUN(5)
      CHARACTER*(*) MESSG
      CHARACTER*20 MESTAB(10),MES
      DIMENSION NERTAB(10),LEVTAB(10),KOUNT(10)
      SAVE MESTAB,NERTAB,LEVTAB,KOUNT,KOUNTX
C    NEXT TWO DATA STATEMENTS ARE NECESSARY TO PROVIDE A BLANK
C    ERROR TABLE INITIALLY
      DATA KOUNT(1),KOUNT(2),KOUNT(3),KOUNT(4),KOUNT(5),
     1     KOUNT(6),KOUNT(7),KOUNT(8),KOUNT(9),KOUNT(10)
     2     /0,0,0,0,0,0,0,0,0,0/
      DATA KOUNTX/0/
C***FIRST EXECUTABLE STATEMENT  XERSAV
      IF (NMESSG.GT.0) GO TO 80
C    DUMP THE TABLE
         IF (KOUNT(1).EQ.0) RETURN
C        PRINT TO EACH UNIT
         CALL XGETUA(LUN,NUNIT)
         DO 60 KUNIT=1,NUNIT
            IUNIT = LUN(KUNIT)
            IF (IUNIT.EQ.0) IUNIT = I1MACH1(4)
C           PRINT TABLE HEADER
            WRITE (IUNIT,10)
   10       FORMAT (32H0          ERROR MESSAGE SUMMARY/
     1      51H MESSAGE START             NERR     LEVEL     COUNT)
C           PRINT BODY OF TABLE
            DO 20 I=1,10
               IF (KOUNT(I).EQ.0) GO TO 30
               WRITE (IUNIT,15) MESTAB(I),NERTAB(I),LEVTAB(I),KOUNT(I)
   15          FORMAT (1X,A20,3I10)
   20       CONTINUE
   30       CONTINUE
C           PRINT NUMBER OF OTHER ERRORS
            IF (KOUNTX.NE.0) WRITE (IUNIT,40) KOUNTX
   40       FORMAT (41H0OTHER ERRORS NOT INDIVIDUALLY TABULATED=,I10)
            WRITE (IUNIT,50)
   50       FORMAT (1X)
   60    CONTINUE
         IF (NMESSG.LT.0) RETURN
C        CLEAR THE ERROR TABLES
         DO 70 I=1,10
   70       KOUNT(I) = 0
         KOUNTX = 0
         RETURN
   80 CONTINUE
C    PROCESS A MESSAGE...
C    SEARCH FOR THIS MESSG, OR ELSE AN EMPTY SLOT FOR THIS MESSG,
C    OR ELSE DETERMINE THAT THE ERROR TABLE IS FULL.
      MES = MESSG
      DO 90 I=1,10
         II = I
         IF (KOUNT(I).EQ.0) GO TO 110
         IF (MES.NE.MESTAB(I)) GO TO 90
         IF (NERR.NE.NERTAB(I)) GO TO 90
```

```
        IF (I.EQ.1) INDEX = 3
        IUNITA(I) = J4SAVE(INDEX,0,.FALSE.)
   30 CONTINUE
      RETURN
      END
      SUBROUTINE FDUMP
C***FIRST EXECUTABLE STATEMENT  FDUMP
      RETURN
      END
      INTEGER FUNCTION I1MACH1(I)
C***BEGIN PROLOGUE  I1MACH
C***DATE WRITTEN   750101   (YYMMDD)
C***REVISION DATE  910131   (YYMMDD)
C***CATEGORY NO.  R1
C***KEYWORDS  MACHINE CONSTANTS
C***AUTHOR  FOX, P. A., (BELL LABS)
C           HALL, A. D., (BELL LABS)
C           SCHRYER, N. L., (BELL LABS)
C***PURPOSE  Returns integer machine dependent constants
C***DESCRIPTION
C
C     This is the CMLIB version of I1MACH, the integer machine
C     constants subroutine originally developed for the PORT library.
C
C     I1MACH can be used to obtain machine-dependent parameters
C     for the local machine environment.  It is a function
C     subroutine with one (input) argument, and can be called
C     as follows, for example
C
C          K = I1MACH(I)
C
C     where I=1,...,16.  The (output) value of K above is
C     determined by the (input) value of I.  The results for
C     various values of I are discussed below.
C
C  I/O unit numbers.
C    I1MACH( 1) = the standard input unit.
C    I1MACH( 2) = the standard output unit.
C    I1MACH( 3) = the standard punch unit.
C    I1MACH( 4) = the standard error message unit.
C
C  Words.
C    I1MACH( 5) = the number of bits per integer storage unit.
C    I1MACH( 6) = the number of characters per integer storage unit.
C
C  Integers.
C    assume integers are represented in the S-digit, base-A form
C
C               sign ( X(S-1)*A**(S-1) + ... + X(1)*A + X(0) )
C
C               where 0 .LE. X(I) .LT. A for I=0,...,S-1.
C    I1MACH( 7) = A, the base.
C    I1MACH( 8) = S, the number of base-A digits.
C    I1MACH( 9) = A**S - 1, the largest magnitude.
C
C  Floating-Point Numbers.
C    Assume floating-point numbers are represented in the T-digit,
C    base-B form
C               sign (B**E)*( (X(1)/B) + ... + (X(T)/B**T) )
C
C               where 0 .LE. X(I) .LT. B for I=1,...,T,
```

```
C            0 .LT. X(1), and EMIN .LE. E .LE. EMAX.
C   I1MACH(10) = B, the base.
C
C  Single-Precision
C   I1MACH(11) = T, the number of base-B digits.
C   I1MACH(12) = EMIN, the smallest exponent E.
C   I1MACH(13) = EMAX, the largest exponent E.
C
C  Double-Precision
C   I1MACH(14) = T, the number of base-B digits.
C   I1MACH(15) = EMIN, the smallest exponent E.
C   I1MACH(16) = EMAX, the largest exponent E.
C
C  To alter this function for a particular environment,
C  the desired set of DATA statements should be activated by
C  removing the C from column 1.  Also, the values of
C  I1MACH(1) - I1MACH(4) should be checked for consistency
C  with the local operating system.
C***REFERENCES  FOX P.A., HALL A.D., SCHRYER N.L.,*FRAMEWORK FOR A
C               PORTABLE LIBRARY*, ACM TRANSACTIONS ON MATHEMATICAL
C               SOFTWARE, VOL. 4, NO. 2, JUNE 1978, PP. 177-188.
C***ROUTINES CALLED  (NONE)
C***END PROLOGUE  I1MACH
C
      INTEGER IMACH(16),OUTPUT
      EQUIVALENCE (IMACH(4),OUTPUT)
C
C     MACHINE CONSTANTS FOR IEEE ARITHMETIC MACHINES, SUCH AS THE AT&T
C     3B SERIES, MOTOROLA 68000 BASED MACHINES (E.G. SUN 3 AND AT&T
C     PC 7300), AND 8087 BASED MICROS (E.G. IBM PC AND AT&T 6300).
C
C === MACHINE = IEEE.MOST-SIG-BYTE-FIRST
C === MACHINE = IEEE.LEAST-SIG-BYTE-FIRST
C === MACHINE = SUN
C === MACHINE = 68000
C === MACHINE = 8087
C === MACHINE = IBM.PC
C === MACHINE = ATT.3B
C === MACHINE = ATT.7300
C === MACHINE = ATT.6300
      DATA IMACH( 1) /    5 /
      DATA IMACH( 2) /    6 /
      DATA IMACH( 3) /    7 /
      DATA IMACH( 4) /    6 /
      DATA IMACH( 5) /   32 /
      DATA IMACH( 6) /    4 /
      DATA IMACH( 7) /    2 /
      DATA IMACH( 8) /   31 /
      DATA IMACH( 9) / 2147483647 /
      DATA IMACH(10) /    2 /
      DATA IMACH(11) /   24 /
      DATA IMACH(12) / -125 /
      DATA IMACH(13) /  128 /
      DATA IMACH(14) /   53 /
      DATA IMACH(15) / -1021 /
      DATA IMACH(16) /  1024 /
C***FIRST EXECUTABLE STATEMENT  I1MACH
      IF (I .LT. 1  .OR.  I .GT. 16)
     1   CALL XERROR1( 'I1MACH -- I OUT OF BOUNDS',25,1,2)
      I1MACH1=IMACH(I)
      RETURN
```

```
      END
      FUNCTION J4SAVE(IWHICH,IVALUE,ISET)
C***BEGIN PROLOGUE  J4SAVE
C***REFER TO  XERROR
C     Abstract
C        J4SAVE saves and recalls several global variables needed
C        by the library error handling routines.
C
C     Description of Parameters
C      --Input--
C        IWHICH - Index of item desired.
C                = 1 Refers to current error number.
C                = 2 Refers to current error control flag.
C                 = 3 Refers to current unit number to which error
C                    messages are to be sent.  (0 means use standard.)
C                 = 4 Refers to the maximum number of times any
C                     message is to be printed (as set by XERMAX).
C                 = 5 Refers to the total number of units to which
C                     each error message is to be written.
C                 = 6 Refers to the 2nd unit for error messages
C                 = 7 Refers to the 3rd unit for error messages
C                 = 8 Refers to the 4th unit for error messages
C                 = 9 Refers to the 5th unit for error messages
C        IVALUE - The value to be set for the IWHICH-th parameter,
C                 if ISET is .TRUE. .
C        ISET   - If ISET=.TRUE., the IWHICH-th parameter will BE
C                 given the value, IVALUE.  If ISET=.FALSE., the
C                 IWHICH-th parameter will be unchanged, and IVALUE
C                 is a dummy parameter.
C      --Output--
C        The (old) value of the IWHICH-th parameter will be returned
C        in the function value, J4SAVE.
C
C    Written by Ron Jones, with SLATEC Common Math Library Subcommittee
C   Adapted from Bell Laboratories PORT Library Error Handler
C    Latest revision ---  23 MAY 1979
C***REFERENCES  JONES R.E., KAHANER D.K., "XERROR, THE SLATEC ERROR-
C                 HANDLING PACKAGE", SAND82-0800, SANDIA LABORATORIES,
C                 1982.
C***ROUTINES CALLED  (NONE)
C***END PROLOGUE  J4SAVE
      LOGICAL ISET
      INTEGER IPARAM(9)
      SAVE IPARAM
      DATA IPARAM(1),IPARAM(2),IPARAM(3),IPARAM(4)/0,2,0,10/
      DATA IPARAM(5)/1/
      DATA IPARAM(6),IPARAM(7),IPARAM(8),IPARAM(9)/0,0,0,0/
C***FIRST EXECUTABLE STATEMENT  J4SAVE
      J4SAVE = IPARAM(IWHICH)
      IF (ISET) IPARAM(IWHICH) = IVALUE
      RETURN
      END
      SUBROUTINE XERABT(MESSG,NMESSG)
C***BEGIN PROLOGUE  XERABT
C***DATE WRITTEN   790801   (YYMMDD)
C***REVISION DATE  820801   (YYMMDD)
C***CATEGORY NO.  R3C
C***KEYWORDS  ERROR,XERROR PACKAGE
C***AUTHOR  JONES, R. E., (SNLA)
C***PURPOSE  Aborts program execution and prints error message.
C***DESCRIPTION
```

```
C   Abstract
C      ***Note*** machine dependent routine
C      XERABT aborts the execution of the program.
C      The error message causing the abort is given in the calling
C      sequence, in case one needs it for printing on a dayfile,
C      for example.
C
C   Description of Parameters
C      MESSG and NMESSG are as in XERROR, except that NMESSG may
C      be zero, in which case no message is being supplied.
C
C   Written by Ron Jones, with SLATEC Common Math Library Subcommittee
C   Latest revision ---  19 MAR 1980
C***REFERENCES  JONES R.E., KAHANER D.K., "XERROR, THE SLATEC ERROR-
C                 HANDLING PACKAGE", SAND82-0800, SANDIA LABORATORIES,
C                 1982.
C***ROUTINES CALLED  (NONE)
C***END PROLOGUE  XERABT
      CHARACTER*(*) MESSG
C***FIRST EXECUTABLE STATEMENT  XERABT
      STOP
      END
      SUBROUTINE XERCTL(MESSG1,NMESSG,NERR,LEVEL,KONTRL)
C***BEGIN PROLOGUE  XERCTL
C***DATE WRITTEN   790801   (YYMMDD)
C***REVISION DATE  820801   (YYMMDD)
C***CATEGORY NO.  R3C
C***KEYWORDS  ERROR,XERROR PACKAGE
C***AUTHOR  JONES, R. E., (SNLA)
C***PURPOSE  Allows user control over handling of individual errors.
C***DESCRIPTION
C   Abstract
C      Allows user control over handling of individual errors.
C      Just after each message is recorded, but before it is
C      processed any further (i.e., before it is printed or
C      a decision to abort is made), a call is made to XERCTL.
C      If the user has provided his own version of XERCTL, he
C      can then override the value of KONTROL used in processing
C      this message by redefining its value.
C      KONTRL may be set to any value from -2 to 2.
C      The meanings for KONTRL are the same as in XSETF, except
C      that the value of KONTRL changes only for this message.
C      If KONTRL is set to a value outside the range from -2 to 2,
C      it will be moved back into that range.
C
C   Description of Parameters
C
C    --Input--
C      MESSG1 - the first word (only) of the error message.
C      NMESSG - same as in the call to XERROR or XERRWV.
C      NERR   - same as in the call to XERROR or XERRWV.
C      LEVEL  - same as in the call to XERROR or XERRWV.
C      KONTRL - the current value of the control flag as set
C               by a call to XSETF.
C
C    --Output--
C      KONTRL - the new value of KONTRL.  If KONTRL is not
C               defined, it will remain at its original value.
C               This changed value of control affects only
C               the current occurrence of the current message.
C***REFERENCES  JONES R.E., KAHANER D.K., "XERROR, THE SLATEC ERROR-
```

```
C             HANDLING PACKAGE", SAND82-0800, SANDIA LABORATORIES,
C             1982.
C***ROUTINES CALLED  (NONE)
C***END PROLOGUE  XERCTL
      CHARACTER*20 MESSG1
C***FIRST EXECUTABLE STATEMENT  XERCTL
      RETURN
      END
      SUBROUTINE XERPRT(MESSG,NMESSG)
C***BEGIN PROLOGUE  XERPRT
C***DATE WRITTEN   790801   (YYMMDD)
C***REVISION DATE  820801   (YYMMDD)
C***CATEGORY NO.  Z
C***KEYWORDS  ERROR,XERROR PACKAGE
C***AUTHOR  JONES, R. E., (SNLA)
C***PURPOSE  Prints error messages.
C***DESCRIPTION
C     Abstract
C        Print the Hollerith message in MESSG, of length NMESSG,
C        on each file indicated by XGETUA.
C     Latest revision ---  19 MAR 1980
C***REFERENCES  JONES R.E., KAHANER D.K., "XERROR, THE SLATEC ERROR-
C             HANDLING PACKAGE", SAND82-0800, SANDIA LABORATORIES,
C             1982.
C***ROUTINES CALLED  I1MACH,S88FMT,XGETUA
C***END PROLOGUE  XERPRT
      INTEGER LUN(5)
      CHARACTER*(*) MESSG
C     OBTAIN UNIT NUMBERS AND WRITE LINE TO EACH UNIT
C***FIRST EXECUTABLE STATEMENT  XERPRT
      CALL XGETUA(LUN,NUNIT)
      LENMES = LEN(MESSG)
      DO 20 KUNIT=1,NUNIT
         IUNIT = LUN(KUNIT)
         IF (IUNIT.EQ.0) IUNIT = I1MACH1(4)
         DO 10 ICHAR=1,LENMES,72
            LAST = MIN0(ICHAR+71 , LENMES)
            WRITE (IUNIT,'(1X,A)') MESSG(ICHAR:LAST)
   10    CONTINUE
   20 CONTINUE
      RETURN
      END
C*****************************************************************************
c □˜FT OLAN AKT˜F TšM DE¦˜□KENLER˜N SONUNA 11 EKLEND˜
* ===========================================================================
* NIST Guide to Available Math Software.
* Fullsource for module BESJ1 from package CMLIB.
* Retrieved from CAMSUN on Wed Feb  4 07:43:52 1998.
* ===========================================================================
      FUNCTION BESJ1(X)
C***BEGIN PROLOGUE  BESJ1
C***DATE WRITTEN   780601   (YYMMDD)
C***REVISION DATE  820801   (YYMMDD)
C***CATEGORY NO.  C10A1
C***KEYWORDS  BESSEL FUNCTION,FIRST KIND,ORDER ONE,SPECIAL FUNCTION
C***AUTHOR  FULLERTON, W., (LANL)
C***PURPOSE  Computes the Bessel function of the first kind of order one
C***DESCRIPTION
C
C BESJ1(X) calculates the Bessel function of the first kind of
C order one for real argument X.
```

```
C
C Series for BJ1       on the interval  0.         to  1.60000D+01
C                              with weighted error   4.48E-17
C                               log weighted error  16.35
C                     significant figures required  15.77
C                          decimal places required  16.89
C
C Series for BM1       on the interval  0.         to  6.25000D-02
C                              with weighted error   5.61E-17
C                               log weighted error  16.25
C                     significant figures required  14.97
C                          decimal places required  16.91
C
C Series for BTH1      on the interval  0.         to  6.25000D-02
C                              with weighted error   4.10E-17
C                               log weighted error  16.39
C                     significant figures required  15.96
C                          decimal places required  17.08
C***REFERENCES  (NONE)
C***ROUTINES CALLED  CSEVL1,INITS,R1MACH11,XERROR
C***END PROLOGUE  BESJ1
      DIMENSION BJ1CS(12), BM1CS(21), BTH1CS(24)
      DATA BJ1 CS( 1) /   -.1172614151 3332787E0 /
      DATA BJ1 CS( 2) /   -.2536152183 0790640E0 /
      DATA BJ1 CS( 3) /    .0501270809 84469569E0 /
      DATA BJ1 CS( 4) /   -.0046315148 09625081E0 /
      DATA BJ1 CS( 5) /    .0002479962 29415914E0 /
      DATA BJ1 CS( 6) /   -.0000086789 48686278E0 /
      DATA BJ1 CS( 7) /    .0000002142 93917143E0 /
      DATA BJ1 CS( 8) /   -.0000000039 36093079E0 /
      DATA BJ1 CS( 9) /    .0000000000 55911823E0 /
      DATA BJ1 CS(10) /   -.0000000000 00632761E0 /
      DATA BJ1 CS(11) /    .0000000000 00005840E0 /
      DATA BJ1 CS(12) /   -.0000000000 00000044E0 /
      DATA BM1 CS( 1) /    .1047362510 931285E0 /
      DATA BM1 CS( 2) /    .0044244389 3702345E0 /
      DATA BM1 CS( 3) /   -.0000566163 9504035E0 /
      DATA BM1 CS( 4) /    .0000023134 9417339E0 /
      DATA BM1 CS( 5) /   -.0000001737 7182007E0 /
      DATA BM1 CS( 6) /    .0000000189 3209930E0 /
      DATA BM1 CS( 7) /   -.0000000026 5416023E0 /
      DATA BM1 CS( 8) /    .0000000004 4740209E0 /
      DATA BM1 CS( 9) /   -.0000000000 8691795E0 /
      DATA BM1 CS(10) /    .0000000000 1891492E0 /
      DATA BM1 CS(11) /   -.0000000000 0451884E0 /
      DATA BM1 CS(12) /    .0000000000 0116765E0 /
      DATA BM1 CS(13) /   -.0000000000 0032265E0 /
      DATA BM1 CS(14) /    .0000000000 0009450E0 /
      DATA BM1 CS(15) /   -.0000000000 0002913E0 /
      DATA BM1 CS(16) /    .0000000000 0000939E0 /
      DATA BM1 CS(17) /   -.0000000000 0000315E0 /
      DATA BM1 CS(18) /    .0000000000 0000109E0 /
      DATA BM1 CS(19) /   -.0000000000 0000039E0 /
      DATA BM1 CS(20) /    .0000000000 0000014E0 /
      DATA BM1 CS(21) /   -.0000000000 0000005E0 /
      DATA BTH1CS( 1) /    .7406014102 6313850E0 /
      DATA BTH1CS( 2) /   -.0045717556 59637690E0 /
      DATA BTH1CS( 3) /    .0001198185 10964326E0 /
      DATA BTH1CS( 4) /   -.0000069645 61891648E0 /
      DATA BTH1CS( 5) /    .0000006554 95621447E0 /
      DATA BTH1CS( 6) /   -.0000000840 66228945E0 /
```

```
      DATA BTH1CS( 7) /    .0000000133 76886564E0 /
      DATA BTH1CS( 8) /   -.0000000024 99565654E0 /
      DATA BTH1CS( 9) /    .0000000005 29495100E0 /
      DATA BTH1CS(10) /   -.0000000001 24135944E0 /
      DATA BTH1CS(11) /    .0000000000 31656485E0 /
      DATA BTH1CS(12) /   -.0000000000 08668640E0 /
      DATA BTH1CS(13) /    .0000000000 02523758E0 /
      DATA BTH1CS(14) /   -.0000000000 00775085E0 /
      DATA BTH1CS(15) /    .0000000000 00249527E0 /
      DATA BTH1CS(16) /   -.0000000000 00083773E0 /
      DATA BTH1CS(17) /    .0000000000 00029205E0 /
      DATA BTH1CS(18) /   -.0000000000 00010534E0 /
      DATA BTH1CS(19) /    .0000000000 00003919E0 /
      DATA BTH1CS(20) /   -.0000000000 00001500E0 /
      DATA BTH1CS(21) /    .0000000000 00000589E0 /
      DATA BTH1CS(22) /   -.0000000000 00000237E0 /
      DATA BTH1CS(23) /    .0000000000 00000097E0 /
      DATA BTH1CS(24) /   -.0000000000 00000040E0 /
      DATA PI4 / 0.7853981633 9744831E0 /
      DATA NTJ1, NTM1, NTTH1, XSML, XMIN, XMAX / 3*0, 3*0./
C***FIRST EXECUTABLE STATEMENT  BESJ1
      IF (NTJ1.NE.0) GO TO 10
      NTJ1 = INITS (BJ1CS, 12, 0.1*R1MACH11(3))
      NTM1 = INITS (BM1CS, 21, 0.1*R1MACH11(3))
      NTTH1 = INITS (BTH1CS, 24, 0.1*R1MACH11(3))
C
      XSML = SQRT (8.0*R1MACH11(3))
      XMIN = 2.0*R1MACH11(1)
      XMAX = 1.0/R1MACH11(4)
C
 10   Y = ABS(X)
      IF (Y.GT.4.0) GO TO 20
C
      BESJ1 = 0.
      IF (Y.EQ.0.0) RETURN
      IF (Y.LT.XMIN) CALL XERROR ( 'BESJ1   ABS(X) SO SMALL J1 UNDERFLOW
     1S', 37, 1, 1)
      IF (Y.GT.XMIN) BESJ1 = 0.5*X
      IF (Y.GT.XSML) BESJ1 = X * (.25 + CSEVL1(.125*Y*Y-1.,BJ1CS,NTJ1))
      RETURN
C
 20   IF (Y.GT.XMAX) CALL XERROR (  'BESJ1   NO PRECISION BECAUSE ABS(X)
     1 IS BIG', 42, 2, 2)
      Z = 32.0/Y**2 - 1.0
      AMPL = (0.75 + CSEVL1(Z, BM1CS, NTM1)) / SQRT(Y)
      THETA = Y - 3.0*PI4 + CSEVL1(Z, BTH1CS, NTTH1) / Y
      BESJ1 = SIGN (AMPL, X) * COS (THETA)
C
      RETURN
      END
      FUNCTION CSEVL1(X,CS,N)
C***BEGIN PROLOGUE  CSEVL1
C***DATE WRITTEN   770401   (YYMMDD)
C***REVISION DATE  820801   (YYMMDD)
C***CATEGORY NO.  C3A2
C***KEYWORDS  CHEBYSHEV,FNLIB,SPECIAL FUNCTION
C***AUTHOR  FULLERTON, W., (LANL)
C***PURPOSE  Evaluate the N-term Chebyshev series CS at X.
C***DESCRIPTION
C
C Evaluate the N-term Chebyshev series CS at X.  Adapted from
```

```
C R. Broucke, Algorithm 446, C.A.C.M., 16, 254 (1973). Also see Fox
C and Parker, Chebyshev Polynomials in Numerical Analysis, Oxford Press,
C page 56.
C
C      Input Arguments --
C X    value at which the series is to be evaluated.
C CS   array of N terms of a Chebyshev series.  In eval-
C      uating CS, only half the first coefficient is summed.
C N    number of terms in array CS.
C***REFERENCES  (NONE)
C***ROUTINES CALLED  XERROR
C***END PROLOGUE  CSEVL1
C
      DIMENSION CS(1)
C***FIRST EXECUTABLE STATEMENT  CSEVL1
      IF(N.LT.1) CALL XERROR( 'CSEVL1  NUMBER OF TERMS LE 0', 28, 2,2)
      IF(N.GT.1000) CALL XERROR ( 'CSEVL1   NUMBER OF TERMS GT 1000',
     1   31,3,2)
      IF (X.LT. -1.0 .OR. X.GT. 1.0) CALL XERROR( 'CSEVL1  X OUTSIDE(-
     11,+1)', 25, 1, 1)
C
      B1=0.
      B0=0.
      TWOX=2.*X
      DO 10 I=1,N
      B2=B1
      B1=B0
      NI=N+1-I
      B0=TWOX*B1-B2+CS(NI)
 10   CONTINUE
C
      CSEVL1= 0.5 * (B0-B2)
C
      RETURN
      END
      FUNCTION INITS1(OS,NOS,ETA)
C***BEGIN PROLOGUE  INITS1
C***DATE WRITTEN   770401   (YYMMDD)
C***REVISION DATE  820801   (YYMMDD)
C***CATEGORY NO.  C3A2
C***KEYWORDS  INITIALIZE,ORTHOGONAL SERIES,SPECIAL FUNCTION
C***AUTHOR  FULLERTON, W., (LANL)
C***PURPOSE  Initializes an orthogonal series so that it defines the
C            number of terms to carry in the series to meet a specified
C            error.
C***DESCRIPTION
C
C Initialize the orthogonal series so that INITS1 is the number of terms
C needed to insure the error is no larger than ETA.  Ordinarily, ETA
C will be chosen to be one-tenth machine precision.
C
C            Input Arguments --
C OS     array of NOS coefficients in an orthogonal series.
C NOS    number of coefficients in OS.
C ETA    requested accuracy of series.
C***REFERENCES  (NONE)
C***ROUTINES CALLED  XERROR
C***END PROLOGUE  INITS1
      DIMENSION OS(NOS)
C***FIRST EXECUTABLE STATEMENT  INITS1
      IF (NOS.LT.1) CALL XERROR ( 'INITS1 NUMBER OF COEFFICIENTS LT 1',
```

```
     1 35, 2, 2)
C
      ERR = 0.
      DO 10 II=1,NOS
        I = NOS + 1 - II
        ERR = ERR + ABS(OS(I))
        IF (ERR.GT.ETA) GO TO 20
 10   CONTINUE
C
 20   IF (I.EQ.NOS) CALL XERROR ( 'INITS1  ETA MAY BE TOO SMALL', 28,
     1  1, 2)
      INITS1= I
C
      RETURN
      END
      REAL FUNCTION R1MACH11(I)
C***BEGIN PROLOGUE  R1MACH11
C***DATE WRITTEN   790101   (YYMMDD)
C***REVISION DATE  910131   (YYMMDD)
C***CATEGORY NO.  R1
C***KEYWORDS  MACHINE CONSTANTS
C***AUTHOR  FOX, P. A., (BELL LABS)
C           HALL, A. D., (BELL LABS)
C           SCHRYER, N. L., (BELL LABS)
C***PURPOSE  Returns single precision machine dependent constants
C***DESCRIPTION
C
C     This is the CMLIB version of R1MACH11, the real machine
C     constants subroutine originally developed for the PORT library.
C
C     R1MACH11 can be used to obtain machine-dependent parameters
C     for the local machine environment.  It is a function
C     subroutine with one (input) argument, and can be called
C     as follows, for example
C
C          A = R1MACH11(I)
C
C     where I=1,...,5.  The (output) value of A above is
C     determined by the (input) value of I.  The results for
C     various values of I are discussed below.
C
C  Single-Precision Machine Constants
C  R1MACH11(1) = B**(EMIN-1), the smallest positive magnitude.
```

```
C
      EQUIVALENCE (RMACH(1),SMALL(1))
      EQUIVALENCE (RMACH(2),LARGE(1))
      EQUIVALENCE (RMACH(3),RIGHT(1))
      EQUIVALENCE (RMACH(4),DIVER(1))
      EQUIVALENCE (RMACH(5),LOG10(1))
C
C
C     MACHINE CONSTANTS FOR IEEE ARITHMETIC MACHINES, SUCH AS THE AT&T
C     3B SERIES, MOTOROLA 68000 BASED MACHINES (E.G. SUN 3 AND AT&T
C     PC 7300), AND 8087 BASED MICROS (E.G. IBM PC AND AT&T 6300).
C
C === MACHINE = IEEE.MOST-SIG-BYTE-FIRST
C === MACHINE = IEEE.LEAST-SIG-BYTE-FIRST
C === MACHINE = SUN
C === MACHINE = 68000
C === MACHINE = 8087
C === MACHINE = IBM.PC
C === MACHINE = ATT.3B
C === MACHINE = ATT.6300
C === MACHINE = ATT.7300
      DATA SMALL(1) /     8388608 /
      DATA LARGE(1) /  2139095039 /
      DATA RIGHT(1) /   864026624 /
      DATA DIVER(1) /   872415232 /
      DATA LOG10(1) /  1050288283 /
C***FIRST EXECUTABLE STATEMENT  R1MACH11
      IF (I .LT. 1  .OR.  I .GT. 5)
     1   CALL XERROR ( 'R1MACH11 -- I OUT OF BOUNDS',25,1,2)
C
      R1MACH11 = RMACH(I)
      RETURN
C
      END
      SUBROUTINE XERROR11(MESSG,NMESSG,NERR,LEVEL)
C***BEGIN PROLOGUE  XERROR11
C***DATE WRITTEN   790801   (YYMMDD)
C***REVISION DATE  820801   (YYMMDD)
C***CATEGORY NO.  R3C
C***KEYWORDS  ERROR,XERROR11 PACKAGE
C***AUTHOR  JONES, R. E., (SNLA)
C***PURPOSE  Processes an error (diagnostic) message.
C***DESCRIPTION
C     Abstract
C        XERROR11 processes a diagnostic message, in a manner
C        determined by the value of LEVEL and the current value
C        of the library error control flag, KONTRL.
C        (See subroutine XSETF for details.)
C
C     Description of Parameters
C      --Input--
C        MESSG - the Hollerith message to be processed, containing
C                no more than 72 characters.
C        NMESSG- the actual number of characters in MESSG.
C        NERR  - the error number associated with this message.
C                NERR must not be zero.
C        LEVEL - error category.
C                =2 means this is an unconditionally fatal error.
C                =1 means this is a recoverable error.  (I.e., it is
C                   non-fatal if XSETF has been appropriately called.)
C                =0 means this is a warning message only.
```

```
C            =-1 means this is a warning message which is to be
C              printed at most once, regardless of how many
C              times this call is executed.
C
C    Examples
C      CALL XERROR11('SMOOTH -- NUM WAS ZERO.',23,1,2)
C      CALL XERROR11('INTEG  -- LESS THAN FULL ACCURACY ACHIEVED.',
C              43,2,1)
C      CALL XERROR11('ROOTER -- ACTUAL ZERO OF F FOUND BEFORE INTERVAL F
C    1ULLY COLLAPSED.',65,3,0)
C      CALL XERROR('EXP    -- UNDERFLOWS BEING SET TO ZERO.',39,1,-1)
C
C    Latest revision ---  19 MAR 1980
C    Written by Ron Jones, with SLATEC Common Math Library Subcommittee
C***REFERENCES  JONES R.E., KAHANER D.K., "XERROR, THE SLATEC ERROR-
C                 HANDLING PACKAGE", SAND82-0800, SANDIA LABORATORIES,
C                 1982.
C***ROUTINES CALLED  XERRWV
C***END PROLOGUE  XERROR
      CHARACTER*(*) MESSG
C***FIRST EXECUTABLE STATEMENT  XERROR
      CALL XERRWV(MESSG,NMESSG,NERR,LEVEL,0,0,0,0,0.,0.)
      RETURN
      END
      SUBROUTINE XERRWV11(MESSG,NMESSG,NERR,LEVEL,NI,I1,I2,NR,R1,R2)
C***BEGIN PROLOGUE  XERRWV11
C***DATE WRITTEN   800319   (YYMMDD)
C***REVISION DATE  820801   (YYMMDD)
C***CATEGORY NO.  R3C
C***KEYWORDS  ERROR,XERROR PACKAGE
C***AUTHOR  JONES, R. E., (SNLA)
C***PURPOSE  Processes error message allowing 2 integer and two real
C            values to be included in the message.
C***DESCRIPTION
C    Abstract
C      XERRWV processes a diagnostic message, in a manner
C      determined by the value of LEVEL and the current value
C      of the library error control flag, KONTRL.
C      (See subroutine XSETF for details.)
C      In addition, up to two integer values and two real
C      values may be printed along with the message.
C
C    Description of Parameters
C     --Input--
C      MESSG - the Hollerith message to be processed.
C      NMESSG- the actual number of characters in MESSG.
C      NERR  - the error number associated with this message.
C              NERR must not be zero.
C      LEVEL - error category.
C              =2 means this is an unconditionally fatal error.
C              =1 means this is a recoverable error.  (I.e., it is
C                 non-fatal if XSETF has been appropriately called.)
C              =0 means this is a warning message only.
C              =-1 means this is a warning message which is to be
C                 printed at most once, regardless of how many
C                 times this call is executed.
C      NI    - number of integer values to be printed. (0 to 2)
C      I1    - first integer value.
C      I2    - second integer value.
C      NR    - number of real values to be printed. (0 to 2)
C      R1    - first real value.
```

```
C       R2    - second real value.
C
C     Examples
C       CALL XERRWV('SMOOTH -- NUM (=I1) WAS ZERO.',29,1,2,
C    1   1,NUM,0,0,0.,0.)
C       CALL XERRWV('QUADXY -- REQUESTED ERROR (R1) LESS THAN MINIMUM (
C    1R2).,54,77,1,0,0,0,2,ERRREQ,ERRMIN)
C
C     Latest revision ---  19 MAR 1980
C     Written by Ron Jones, with SLATEC Common Math Library Subcommittee
C***REFERENCES  JONES R.E., KAHANER D.K., "XERROR, THE SLATEC ERROR-
C                 HANDLING PACKAGE", SAND82-0800, SANDIA LABORATORIES,
C                 1982.
C***ROUTINES CALLED  FDUMP,I1MACH,J4SAVE,XERABT,XERCTL,XERPRT,XERSAV,
C                    XGETUA
C***END PROLOGUE  XERRWV
      CHARACTER*(*) MESSG
      CHARACTER*20 LFIRST
      CHARACTER*37 FORM
      DIMENSION LUN(5)
C     GET FLAGS
C***FIRST EXECUTABLE STATEMENT  XERRWV
      LKNTRL = J4SAVE11(2,0,.FALSE.)
      MAXMES = J4SAVE11(4,0,.FALSE.)
C     CHECK FOR VALID INPUT
      IF ((NMESSG.GT.0).AND.(NERR.NE.0).AND.
     1    (LEVEL.GE.(-1)).AND.(LEVEL.LE.2)) GO TO 10
         IF (LKNTRL.GT.0) CALL XERPRT11('FATAL ERROR IN...',17)
         CALL XERPRT11('XERROR11 -- INVALID INPUT',23)
         IF (LKNTRL.GT.0) CALL FDUMP11
         IF (LKNTRL.GT.0) CALL XERPRT11('JOB ABORT DUE TO FATAL ERROR.',
     1  29)
         IF (LKNTRL.GT.0) CALL XERSAV11(' ',0,0,0,KDUMMY)
         CALL XERABT11('XERROR11 -- INVALID INPUT',23)
         RETURN
   10 CONTINUE
C     RECORD MESSAGE
      JUNK = J4SAVE11(1,NERR,.TRUE.)
      CALL XERSAV11(MESSG,NMESSG,NERR,LEVEL,KOUNT)
C     LET USER OVERRIDE
      LFIRST = MESSG
      LMESSG = NMESSG
      LERR = NERR
      LLEVEL = LEVEL
      CALL XERCTL11(LFIRST,LMESSG,LERR,LLEVEL,LKNTRL)
C     RESET TO ORIGINAL VALUES
      LMESSG = NMESSG
      LERR = NERR
      LLEVEL = LEVEL
      LKNTRL = MAX0(-2,MIN0(2,LKNTRL))
      MKNTRL = IABS(LKNTRL)
C     DECIDE WHETHER TO PRINT MESSAGE
      IF ((LLEVEL.LT.2).AND.(LKNTRL.EQ.0)) GO TO 100
      IF (((LLEVEL.EQ.(-1)).AND.(KOUNT.GT.MIN0(1,MAXMES)))
     1.OR.((LLEVEL.EQ.0)   .AND.(KOUNT.GT.MAXMES))
     2.OR.((LLEVEL.EQ.1)   .AND.(KOUNT.GT.MAXMES).AND.(MKNTRL.EQ.1))
     3.OR.((LLEVEL.EQ.2)   .AND.(KOUNT.GT.MAX0(1,MAXMES)))) GO TO 100
         IF (LKNTRL.LE.0) GO TO 20
            CALL XERPRT11(' ',1)
C           INTRODUCTION
            IF (LLEVEL.EQ.(-1)) CALL XERPRT11
```

```
     1('WARNING MESSAGE...THIS MESSAGE WILL ONLY BE PRINTED ONCE.',57)
         IF (LLEVEL.EQ.0) CALL XERPRT11('WARNING IN...',13)
         IF (LLEVEL.EQ.1) CALL XERPRT11
     1    ('RECOVERABLE ERROR IN...',23)
         IF (LLEVEL.EQ.2) CALL XERPRT11('FATAL ERROR IN...',17)
   20  CONTINUE
C      MESSAGE
       CALL XERPRT11(MESSG,LMESSG)
       CALL XGETUA11(LUN,NUNIT)
       ISIZEI = LOG10(FLOAT(I1MACH11(9))) + 1.0
       ISIZEF = LOG10(FLOAT(I1MACH11(10))**I1MACH11(11)) + 1.0
       DO 50 KUNIT=1,NUNIT
         IUNIT = LUN(KUNIT)
         IF (IUNIT.EQ.0) IUNIT = I1MACH11(4)
         DO 22 I=1,MIN(NI,2)
           WRITE (FORM,21) I,ISIZEI
   21        FORMAT ('(11X,21HIN ABOVE MESSAGE, I',I1,'=,I',I2,')   ')
           IF (I.EQ.1) WRITE (IUNIT,FORM) I1
           IF (I.EQ.2) WRITE (IUNIT,FORM) I2
   22    CONTINUE
         DO 24 I=1,MIN(NR,2)
           WRITE (FORM,23) I,ISIZEF+10,ISIZEF
   23        FORMAT ('(11X,21HIN ABOVE MESSAGE, R',I1,'=,E',
     1       I2,'.',I2,')')
           IF (I.EQ.1) WRITE (IUNIT,FORM) R1
           IF (I.EQ.2) WRITE (IUNIT,FORM) R2
   24    CONTINUE
         IF (LKNTRL.LE.0) GO TO 40
C            ERROR NUMBER
           WRITE (IUNIT,30) LERR
   30        FORMAT (15H ERROR NUMBER =,I10)
   40    CONTINUE
   50  CONTINUE
C      TRACE-BACK
       IF (LKNTRL.GT.0) CALL FDUMP11
  100 CONTINUE
     IFATAL = 0
     IF ((LLEVEL.EQ.2).OR.((LLEVEL.EQ.1).AND.(MKNTRL.EQ.2)))
    1IFATAL = 1
C    QUIT HERE IF MESSAGE IS NOT FATAL
     IF (IFATAL.LE.0) RETURN
     IF ((LKNTRL.LE.0).OR.(KOUNT.GT.MAX0(1,MAXMES))) GO TO 120
C      PRINT REASON FOR ABORT
       IF (LLEVEL.EQ.1) CALL XERPRT11
    1  ('JOB ABORT DUE TO UNRECOVERED ERROR.',35)
       IF (LLEVEL.EQ.2) CALL XERPRT11
    1  ('JOB ABORT DUE TO FATAL ERROR.',29)
C      PRINT ERROR SUMMARY
       CALL XERSAV11(' ',-1,0,0,KDUMMY)
  120 CONTINUE
C    ABORT
     IF ((LLEVEL.EQ.2).AND.(KOUNT.GT.MAX0(1,MAXMES))) LMESSG = 0
     CALL XERABT11(MESSG,LMESSG)
     RETURN
     END
     SUBROUTINE XERSAV11(MESSG,NMESSG,NERR,LEVEL,ICOUNT)
C***BEGIN PROLOGUE  XERSAV
C***DATE WRITTEN   800319   (YYMMDD)
C***REVISION DATE  820801   (YYMMDD)
C***CATEGORY NO.  Z
C***KEYWORDS  ERROR,XERROR PACKAGE
```

```
C***AUTHOR  JONES, R. E., (SNLA)
C***PURPOSE  Records that an error occurred.
C***DESCRIPTION
C     Abstract
C        Record that this error occurred.
C
C     Description of Parameters
C     --Input--
C       MESSG, NMESSG, NERR, LEVEL are as in XERROR,
C       except that when NMESSG=0 the tables will be
C       dumped and cleared, and when NMESSG is less than zero the
C       tables will be dumped and not cleared.
C     --Output--
C       ICOUNT will be the number of times this message has
C       been seen, or zero if the table has overflowed and
C       does not contain this message specifically.
C       When NMESSG=0, ICOUNT will not be altered.
C
C     Written by Ron Jones, with SLATEC Common Math Library Subcommittee
C     Latest revision ---  19 Mar 1980
C***REFERENCES  JONES R.E., KAHANER D.K., "XERROR, THE SLATEC ERROR-
C                 HANDLING PACKAGE", SAND82-0800, SANDIA LABORATORIES,
C                 1982.
C***ROUTINES CALLED  I1MACH,S88FMT,XGETUA
C***END PROLOGUE  XERSAV
      INTEGER LUN(5)
      CHARACTER*(*) MESSG
      CHARACTER*20 MESTAB(10),MES
      DIMENSION NERTAB(10),LEVTAB(10),KOUNT(10)
      SAVE MESTAB,NERTAB,LEVTAB,KOUNT,KOUNTX
C     NEXT TWO DATA STATEMENTS ARE NECESSARY TO PROVIDE A BLANK
C     ERROR TABLE INITIALLY
      DATA KOUNT(1),KOUNT(2),KOUNT(3),KOUNT(4),KOUNT(5),
     1     KOUNT(6),KOUNT(7),KOUNT(8),KOUNT(9),KOUNT(10)
     2     /0,0,0,0,0,0,0,0,0,0/
      DATA KOUNTX/0/
C***FIRST EXECUTABLE STATEMENT  XERSAV
      IF (NMESSG.GT.0) GO TO 80
C     DUMP THE TABLE
         IF (KOUNT(1).EQ.0) RETURN
C        PRINT TO EACH UNIT
         CALL XGETUA11(LUN,NUNIT)
         DO 60 KUNIT=1,NUNIT
            IUNIT = LUN(KUNIT)
            IF (IUNIT.EQ.0) IUNIT = I1MACH11(4)
C           PRINT TABLE HEADER
            WRITE (IUNIT,10)
   10       FORMAT (32H0          ERROR MESSAGE SUMMARY/
     1      51H MESSAGE START             NERR     LEVEL     COUNT)
C           PRINT BODY OF TABLE
            DO 20 I=1,10
               IF (KOUNT(I).EQ.0) GO TO 30
               WRITE (IUNIT,15) MESTAB(I),NERTAB(I),LEVTAB(I),KOUNT(I)
   15          FORMAT (1X,A20,3I10)
   20       CONTINUE
   30       CONTINUE
C           PRINT NUMBER OF OTHER ERRORS
            IF (KOUNTX.NE.0) WRITE (IUNIT,40) KOUNTX
   40       FORMAT (41H0OTHER ERRORS NOT INDIVIDUALLY TABULATED=,I10)
            WRITE (IUNIT,50)
   50       FORMAT (1X)
```

```
  60   CONTINUE
       IF (NMESSG.LT.0) RETURN
C       CLEAR THE ERROR TABLES
       DO 70 I=1,10
  70      KOUNT(I) = 0
       KOUNTX = 0
       RETURN
  80 CONTINUE
C     PROCESS A MESSAGE...
C     SEARCH FOR THIS MESSG, OR ELSE AN EMPTY SLOT FOR THIS MESSG,
C     OR ELSE DETERMINE THAT THE ERROR TABLE IS FULL.
      MES = MESSG
      DO 90 I=1,10
       II = I
       IF (KOUNT(I).EQ.0) GO TO 110
       IF (MES.NE.MESTAB(I)) GO TO 90
       IF (NERR.NE.NERTAB(I)) GO TO 90
       IF (LEVEL.NE.LEVTAB(I)) GO TO 90
       GO TO 100
  90 CONTINUE
C     THREE POSSIBLE CASES...
C     TABLE IS FULL
       KOUNTX = KOUNTX+1
       ICOUNT = 1
       RETURN
C     MESSAGE FOUND IN TABLE
 100    KOUNT(II) = KOUNT(II) + 1
       ICOUNT = KOUNT(II)
       RETURN
C     EMPTY SLOT FOUND FOR NEW MESSAGE
 110    MESTAB(II) = MES
       NERTAB(II) = NERR
       LEVTAB(II) = LEVEL
       KOUNT(II)  = 1
       ICOUNT = 1
       RETURN
      END
      SUBROUTINE XGETUA11(IUNITA,N)
C***BEGIN PROLOGUE  XGETUA
C***DATE WRITTEN   790801   (YYMMDD)
C***REVISION DATE  820801   (YYMMDD)
C***CATEGORY NO.  R3C
C***KEYWORDS  ERROR,XERROR PACKAGE
C***AUTHOR  JONES, R. E., (SNLA)
C***PURPOSE  Returns unit number(s) to which error messages are being
C            sent.
C***DESCRIPTION
C     Abstract
C        XGETUA may be called to determine the unit number or numbers
C        to which error messages are being sent.
C        These unit numbers may have been set by a call to XSETUN,
C        or a call to XSETUA, or may be a default value.
C
C     Description of Parameters
C      --Output--
C        IUNIT - an array of one to five unit numbers, depending
C                on the value of N.  A value of zero refers to the
C                default unit, as defined by the I1MACH machine
C                constant routine.  Only IUNIT(1),...,IUNIT(N) are
C                defined by XGETUA.  The values of IUNIT(N+1),...,
C                IUNIT(5) are not defined (for N .LT. 5) or altered
```

```
C           in any way by XGETUA.
C       N    - the number of units to which copies of the
C           error messages are being sent.  N will be in the
C           range from 1 to 5.
C
C   Latest revision --- 19 MAR 1980
C   Written by Ron Jones, with SLATEC Common Math Library Subcommittee
C***REFERENCES  JONES R.E., KAHANER D.K., "XERROR, THE SLATEC ERROR-
C             HANDLING PACKAGE", SAND82-0800, SANDIA LABORATORIES,
C             1982.
C***ROUTINES CALLED  J4SAVE
C***END PROLOGUE  XGETUA
      DIMENSION IUNITA(5)
C***FIRST EXECUTABLE STATEMENT  XGETUA
      N = J4SAVE11(5,0,.FALSE.)
      DO 30 I=1,N
        INDEX = I+4
        IF (I.EQ.1) INDEX = 3
        IUNITA(I) = J4SAVE11(INDEX,0,.FALSE.)
   30 CONTINUE
      RETURN
      END
      SUBROUTINE FDUMP11
C***DESCRIPTION
C       ***Note*** Machine Dependent Routine
C       FDUMP is intended to be replaced by a locally written
C       version which produces a symbolic dump.  Failing this,
C       it should be replaced by a version which prints the
C       subprogram nesting list.  Note that this dump must be
C       printed on each of up to five files, as indicated by the
C       XGETUA routine.  See XSETUA and XGETUA for details.
C
C   Written by Ron Jones, with SLATEC Common Math Library Subcommittee
C   Latest revision --- 23 May 1979
C***ROUTINES CALLED  (NONE)
C***END PROLOGUE  FDUMP
C***FIRST EXECUTABLE STATEMENT  FDUMP
      RETURN
      END
      INTEGER FUNCTION I1MACH11(I)
C***BEGIN PROLOGUE  I1MACH
C***DATE WRITTEN   750101   (YYMMDD)
C***REVISION DATE  910131   (YYMMDD)
C***CATEGORY NO.  R1
C***KEYWORDS  MACHINE CONSTANTS
C***AUTHOR  FOX, P. A., (BELL LABS)
C           HALL, A. D., (BELL LABS)
C           SCHRYER, N. L., (BELL LABS)
C***PURPOSE  Returns integer machine dependent constants
C***DESCRIPTION
C
C   This is the CMLIB version of I1MACH, the integer machine
C   constants subroutine originally developed for the PORT library.
C
C   I1MACH can be used to obtain machine-dependent parameters
C   for the local machine environment.  It is a function
C   subroutine with one (input) argument, and can be called
C   as follows, for example
C
C        K = I1MACH(I)
C
```

```
C    where I=1,...,16.  The (output) value of K above is
C    determined by the (input) value of I.  The results for
C    various values of I are discussed below.
C
C  I/O unit numbers.
C    I1MACH( 1) = the standard input unit.
C    I1MACH( 2) = the standard output unit.
C    I1MACH( 3) = the standard punch unit.
C    I1MACH( 4) = the standard error message unit.
C  Words.
C    I1MACH( 5) = the number of bits per integer storage unit.
C    I1MACH( 6) = the number of characters per integer storage unit.
C  Integers.
C    assume integers are represented in the S-digit, base-A form
C
C               sign ( X(S-1)*A**(S-1) + ... + X(1)*A + X(0) )
C
C               where 0 .LE. X(I) .LT. A for I=0,...,S-1.
C    I1MACH( 7) = A, the base.
C    I1MACH( 8) = S, the number of base-A digits.
C    I1MACH( 9) = A**S - 1, the largest magnitude.
C
C  Floating-Point Numbers.
C    Assume floating-point numbers are represented in the T-digit,
C    base-B form
C               sign (B**E)*( (X(1)/B) + ... + (X(T)/B**T) )
C
C               where 0 .LE. X(I) .LT. B for I=1,...,T,
C               0 .LT. X(1), and EMIN .LE. E .LE. EMAX.
C    I1MACH(10) = B, the base.
C
C  Single-Precision
C    I1MACH(11) = T, the number of base-B digits.
C    I1MACH(12) = EMIN, the smallest exponent E.
C    I1MACH(13) = EMAX, the largest exponent E.
C
C  Double-Precision
C    I1MACH(14) = T, the number of base-B digits.
C    I1MACH(15) = EMIN, the smallest exponent E.
C    I1MACH(16) = EMAX, the largest exponent E.
C
C  To alter this function for a particular environment,
C  the desired set of DATA statements should be activated by
C  removing the C from column 1.  Also, the values of
C  I1MACH(1) - I1MACH(4) should be checked for consistency
C  with the local operating system.
C***REFERENCES  FOX P.A., HALL A.D., SCHRYER N.L.,*FRAMEWORK FOR A
C                 PORTABLE LIBRARY*, ACM TRANSACTIONS ON MATHEMATICAL
C                 SOFTWARE, VOL. 4, NO. 2, JUNE 1978, PP. 177-188.
C***ROUTINES CALLED  (NONE)
C***END PROLOGUE  I1MACH
C
      INTEGER IMACH(16),OUTPUT
      EQUIVALENCE (IMACH(4),OUTPUT)
C
C     MACHINE CONSTANTS FOR IEEE ARITHMETIC MACHINES, SUCH AS THE AT&T
C     3B SERIES, MOTOROLA 68000 BASED MACHINES (E.G. SUN 3 AND AT&T
C     PC 7300), AND 8087 BASED MICROS (E.G. IBM PC AND AT&T 6300).
C
C === MACHINE = IEEE.MOST-SIG-BYTE-FIRST
C === MACHINE = IEEE.LEAST-SIG-BYTE-FIRST
```

```
C === MACHINE = SUN
C === MACHINE = 68000
C === MACHINE = 8087
C === MACHINE = IBM.PC
C === MACHINE = ATT.3B
C === MACHINE = ATT.7300
C === MACHINE = ATT.6300
      DATA IMACH( 1) /    5 /
      DATA IMACH( 2) /    6 /
      DATA IMACH( 3) /    7 /
      DATA IMACH( 4) /    6 /
      DATA IMACH( 5) /   32 /
      DATA IMACH( 6) /    4 /
      DATA IMACH( 7) /    2 /
      DATA IMACH( 8) /   31 /
      DATA IMACH( 9) / 2147483647 /
      DATA IMACH(10) /    2 /
      DATA IMACH(11) /   24 /
      DATA IMACH(12) / -125 /
      DATA IMACH(13) /  128 /
      DATA IMACH(14) /   53 /
      DATA IMACH(15) / -1021 /
      DATA IMACH(16) /  1024 /
C***FIRST EXECUTABLE STATEMENT  I1MACH
     IF (I .LT. 1  .OR.  I .GT. 16)
    1   CALL XERROR11( 'I1MACH11 -- I OUT OF BOUNDS',25,1,2)
C
     I1MACH11=IMACH(I)
     RETURN
C
     END
     FUNCTION J4SAVE11(IWHICH,IVALUE,ISET)
C***BEGIN PROLOGUE  J4SAVE
C***REFER TO  XERROR
C     Abstract
C        J4SAVE saves and recalls several global variables needed
C        by the library error handling routines.
C
C     Description of Parameters
C      --Input--
C        IWHICH - Index of item desired.
C                = 1 Refers to current error number.
C                = 2 Refers to current error control flag.
C                 = 3 Refers to current unit number to which error
C                    messages are to be sent.  (0 means use standard.)
C                 = 4 Refers to the maximum number of times any
C                     message is to be printed (as set by XERMAX).
C                 = 5 Refers to the total number of units to which
C                     each error message is to be written.
C                 = 6 Refers to the 2nd unit for error messages
C                 = 7 Refers to the 3rd unit for error messages
C                 = 8 Refers to the 4th unit for error messages
C                 = 9 Refers to the 5th unit for error messages
C        IVALUE - The value to be set for the IWHICH-th parameter,
C                 if ISET is .TRUE. .
C        ISET   - If ISET=.TRUE., the IWHICH-th parameter will BE
C                 given the value, IVALUE.  If ISET=.FALSE., the
C                 IWHICH-th parameter will be unchanged, and IVALUE
C                 is a dummy parameter.
C      --Output--
C        The (old) value of the IWHICH-th parameter will be returned
```

```
C        in the function value, J4SAVE.
C
C     Written by Ron Jones, with SLATEC Common Math Library Subcommittee
C    Adapted from Bell Laboratories PORT Library Error Handler
C     Latest revision ---  23 MAY 1979
C***REFERENCES  JONES R.E., KAHANER D.K., "XERROR, THE SLATEC ERROR-
C                 HANDLING PACKAGE", SAND82-0800, SANDIA LABORATORIES,
C                 1982.
C***ROUTINES CALLED  (NONE)
C***END PROLOGUE  J4SAVE
      LOGICAL ISET
      INTEGER IPARAM(9)
      SAVE IPARAM
      DATA IPARAM(1),IPARAM(2),IPARAM(3),IPARAM(4)/0,2,0,10/
      DATA IPARAM(5)/1/
      DATA IPARAM(6),IPARAM(7),IPARAM(8),IPARAM(9)/0,0,0,0/
C***FIRST EXECUTABLE STATEMENT  J4SAVE
      J4SAVE11= IPARAM(IWHICH)
      IF (ISET) IPARAM(IWHICH) = IVALUE
      RETURN
      END
      SUBROUTINE XERABT11(MESSG,NMESSG)
C***BEGIN PROLOGUE  XERABT
C***DATE WRITTEN   790801   (YYMMDD)
C***REVISION DATE  820801   (YYMMDD)
C***CATEGORY NO.  R3C
C***KEYWORDS  ERROR,XERROR PACKAGE
C***AUTHOR  JONES, R. E., (SNLA)
C***PURPOSE  Aborts program execution and prints error message.
C***DESCRIPTION
C     Abstract
C        ***Note*** machine dependent routine
C        XERABT aborts the execution of the program.
C        The error message causing the abort is given in the calling
C        sequence, in case one needs it for printing on a dayfile,
C        for example.
C
C     Description of Parameters
C        MESSG and NMESSG are as in XERROR, except that NMESSG may
C        be zero, in which case no message is being supplied.
C
C     Written by Ron Jones, with SLATEC Common Math Library Subcommittee
C     Latest revision ---  19 MAR 1980
C***REFERENCES  JONES R.E., KAHANER D.K., "XERROR, THE SLATEC ERROR-
C                 HANDLING PACKAGE", SAND82-0800, SANDIA LABORATORIES,
C                 1982.
C***ROUTINES CALLED  (NONE)
C***END PROLOGUE  XERABT
      CHARACTER*(*) MESSG
C***FIRST EXECUTABLE STATEMENT  XERABT
      STOP
      END
      SUBROUTINE XERCTL11(MESSG1,NMESSG,NERR,LEVEL,KONTRL)
C***BEGIN PROLOGUE  XERCTL
C***DATE WRITTEN   790801   (YYMMDD)
C***REVISION DATE  820801   (YYMMDD)
C***CATEGORY NO.  R3C
C***KEYWORDS  ERROR,XERROR PACKAGE
C***AUTHOR  JONES, R. E., (SNLA)
C***PURPOSE  Allows user control over handling of individual errors.
C***DESCRIPTION
```

```
C   Abstract
C      Allows user control over handling of individual errors.
C      Just after each message is recorded, but before it is
C      processed any further (i.e., before it is printed or
C      a decision to abort is made), a call is made to XERCTL.
C      If the user has provided his own version of XERCTL, he
C      can then override the value of KONTROL used in processing
C      this message by redefining its value.
C      KONTRL may be set to any value from -2 to 2.
C      The meanings for KONTRL are the same as in XSETF, except
C      that the value of KONTRL changes only for this message.
C      If KONTRL is set to a value outside the range from -2 to 2,
C      it will be moved back into that range.
C
C   Description of Parameters
C
C    --Input--
C      MESSG1 - the first word (only) of the error message.
C      NMESSG - same as in the call to XERROR or XERRWV.
C      NERR   - same as in the call to XERROR or XERRWV.
C      LEVEL  - same as in the call to XERROR or XERRWV.
C      KONTRL - the current value of the control flag as set
C               by a call to XSETF.
C
C    --Output--
C      KONTRL - the new value of KONTRL.  If KONTRL is not
C               defined, it will remain at its original value.
C               This changed value of control affects only
C               the current occurrence of the current message.
C***REFERENCES  JONES R.E., KAHANER D.K., "XERROR, THE SLATEC ERROR-
C               HANDLING PACKAGE", SAND82-0800, SANDIA LABORATORIES,
C               1982.
C***ROUTINES CALLED  (NONE)
C***END PROLOGUE  XERCTL
      CHARACTER*20 MESSG1
C***FIRST EXECUTABLE STATEMENT  XERCTL
      RETURN
      END
      SUBROUTINE XERPRT11(MESSG,NMESSG)
C      Print the Hollerith message in MESSG, of length NMESSG,
C      on each file indicated by XGETUA.
C   Latest revision ---  19 MAR 1980
C***REFERENCES  JONES R.E., KAHANER D.K., "XERROR, THE SLATEC ERROR-
C               HANDLING PACKAGE", SAND82-0800, SANDIA LABORATORIES,
C               1982.
C***ROUTINES CALLED  I1MACH,S88FMT,XGETUA
C***END PROLOGUE  XERPRT
      INTEGER LUN(5)
      CHARACTER*(*) MESSG
C   OBTAIN UNIT NUMBERS AND WRITE LINE TO EACH UNIT
C***FIRST EXECUTABLE STATEMENT  XERPRT
      CALL XGETUA11(LUN,NUNIT)
      LENMES = LEN(MESSG)
      DO 20 KUNIT=1,NUNIT
         IUNIT = LUN(KUNIT)
         IF (IUNIT.EQ.0) IUNIT = I1MACH11(4)
         DO 10 ICHAR=1,LENMES,72
            LAST = MIN0(ICHAR+71 , LENMES)
            WRITE (IUNIT,'(1X,A)') MESSG(ICHAR:LAST)
   10    CONTINUE
   20 CONTINUE
```

```
      RETURN
      END
C*******************************************************************************************
C*   PROGRAM FOR TRANSIENT HEAT CONDUCTION OF A SPHERICAL TANK FOR
C*  LONG TERM HOT WATER STORAGE & USAGE. THE PROBLEM REQUIRES THAT
C*   WALL TEMPERATURE VARIATION OF THE SPHERE TANK WITH TIME.
C*******************************************************************************************
      REAL QN(12),FBAR(12),IHM(5,12),XCB(12),TALM(12),NOH,TNEW(12),Tb(5)
      REAL HH(12),DLTA(12),RBA(12),HT(12),RT(12),RN(12),HO(12),KTM(12)
     $,AA(12),BB(12),CC(12),R(12),REF(12),Ta(5,12),Tair(60),Hrad(60)
      CHARACTER*33,SL(5),TYP(10),City(5)
      DATA REF/.2,.2,.2,.2,.2,.2,.3,.7,.5,.3,.2,.2/
      DATA Tair/23.1,23.3,18.4,12.9,7.7,2.5,0.3,1.,4.7,11.1,16.1,20.,
     E27.2,27.,22.,14.8,7.8,1.5,-1.3,0.,4.7,11.8,17.4,22.9,
     G27.1,26.9,22.2,15.3,9.4,4.5,2.6,3.8,7.2,12.7,18.2,23.7,
     I23.2,23.3,19.7,15.5,11.9,8.,5.1,5.5,6.7,10.9,15.8,20.6,
     i27.5,27.,22.7,18.3,13.9,10.4,8.2,8.8,11.1,15.3,20.1,24.7/
      Data Hrad/23.7,22.3,17.6,12.5,7.4,4.9,6.,8.5,12.7,17.9,20.9,22.1,
     E20.9,19.9,15.6,10.2,5.9,4.6,4.9,8.,10.9,15.3,17.2,21.1,
     G22.1,21.1,16.8,11.7,7.9,5.7,6.2,9.2,12.6,17.7,18.3,22.6,
     I21.5,19.8,14.9,10.,6.7,4.5,4.4,7.4,11.6,16.2,18.9,21.3,
     i19.,17.6,13.9,10.1,6.4,4.4,5.1,7.1,10.3,14.4,19.1,18.3/
      DATA SL/'COARSE GRAVELED','CLAY','SAND','GRANIT','LIMESTONE'/
      DATA TYP/'ONE GLASS COVER BLACK SURFACE','TWO GLASS COVER BLACK
     $ SURFACE','ONE GLASS COVER SELECTIVE SURFACE','TWO GLASS COVER
     $SELECTIVE SURFACE','ENTER RELAXATION NUMBER','ENTER RADIUS OF STOR
     &E','ENTER NUMBER OF HOUSE','ENTER TYPE OF SOIL','Enter Collector
     &Area','Enter Collector Slope'/
      Data City/'Ankara','Elazig','Gaziantep','Istanbul','izmir'/
      Data Tb/40.,38.7,37.1,40.,38.5/
      i1=0
      Do 23 i=1,5
      Do 22 j=1,12
      i1=i1+1
      Ta(i,j)=Tair(i1)
      Ihm(i,j)=Hrad(i1)
      If(j.Eq.12) go to 23
   22 Continue
   23 Continue
    5 WRITE(6,33)City(1),City(2),City(3),City(4),City(5)
   33 FORMAT(6(/),13X,50('*'),/13X,'*',48(' '),'*',/,
     $13X,'*',4X,'If City is ',A9,' PRINT ---> 1    ',7X,'*',/,
     $13X,'*',4X,'           ',A9,' PRINT ---> 2    ',7X,'*',/,
     $13X,'*',4X,'           ',A9,' PRINT ---> 3    ',7X,'*',/,
     $13X,'*',4X,'           ',A9,' PRINT ---> 4    ',7X,'*',/,
     $13X,'*',4X,'           ',A9,' PRINT ---> 5    ',7X,'*',/,
     $13X,'*',7X,'     NOT ANY ONE, PRINT ---> other',7X,'*',/,
     $13X,'*',48(' '),'*',/,13X,50('*'))
      READ(5,*)il
      IF(il.GT.5)GO TO 12
      WRITE(6,7)TYP(9)
      READ(5,*) Ac
      WRITE(6,7)TYP(10)
c     READ(5,*) Tb(il)
C*******************************************************************************************
      CALL COLRAD(il,Tb,HH,DLTA,RBA,HT,RT,RN,HO,KTM,AA,BB,CC,R,Ref,IHM)
C*******************************************************************************************
      WRITE(6,67)Typ(1),Typ(2),Typ(3),Typ(4)
   67 FORMAT(12(/),8X,61('*'),/8X,'*',23X,'OPTIONS',29X,'*',/,
     $8X,'*',4X,'FOR ',A33,', PRINT ---> 1 ',3X,'*',/,
     $8X,'*',4X,'FOR ',A33,', PRINT ---> 2 ',3X,'*',/,
```

```
     $8X,'*',4X,'FOR ',A33,', PRINT ---> 3 ',3X,'*',/,
     $8X,'*',4X,'FOR ',A33,', PRINT ---> 4 ',3X,'*',/,
     $8X,'*',30X,'NOT ANY ONE, PRINT ---> 5 ',3X,'*',/,
     $8X,'*',59(' '),'*',/8X,61('*'),4(/))
      READ(5,*)M
c      M=1
      IF(M.EQ.5) GO TO 12
C****************************************************************************************************
      CALL TAGALFA(il,Tb,Ac,Ref,HH,DLTA,RBA,M,TALM,FR)
C****************************************************************************************************
      WRITE(6,7)Typ(5)
    7 FORMAT(15(/),20X,33('*'),/20X,'*',31(' '),'*',/20X,'*',4X,A23
     $,4X,'*',/20X,'*',31(' '),'*',/20X,33('*'),5(/))
C       READ(5,*)RL
C   Rl:Relaxiation Constant
      Rl=.01
      WRITE(6,7)TYP(6)
      READ(5,*) A
c       A=23.2
c      WRITE(6,7)TYP(7)
c      READ(5,*) NOH
      Noh=1
      WRITE(6,7)TYP(8)
      WRITE(6,11)sl(1),Sl(2),Sl(3),Sl(4),Sl(5)
   11 FORMAT(13X,50('*'),/13X,'*',48(' '),'*',/,
     $13X,'*',4X,'IF SOIL IS',A6,'PRINT ---> 1    ',12X,'*',/,
     $13X,'*',4X,'          ',A6,'PRINT ---> 2    ',12X,'*',/,
     $13X,'*',4X,'          ',A6,'PRINT ---> 3    ',12X,'*',/,
     $13X,'*',4X,'          ',A6,'PRINT ---> 4    ',12X,'*',/,
     $13X,'*',4X,'          ',A6,'PRINT ---> 5    ',12X,'*',/,
     $13X,'*',9X,'   NOT ANY ONE, PRINT ---> other',7X,'*',/,
     $13X,'*',48(' '),'*',/,13X,50('*'))
      READ(5,*)K
c      K=5
      IF(K.GT.5)GO TO 12
C****************************************************************************************************
      CALL SIPSTORE(HT,RT,RN,HO,KTM,AA,BB,CC,Ac,R,Ta,RL,A,NOH,
     $FR,il,K,M,XCB,FBAR,QN,QSY,WY,QLT,QLY,QY,R1,R2,R4,R5,TALM,TNEW)
C****************************************************************************************************
      CALL RESULT(Tb,Ac,Ta,il,M,K,A,NOH,IHM
     $,KTM,R,RN,XCB,FBAR,QN,TALM,TNEW,QSY,WY,QLT,QLY,QY,R1,R2,R4,R5)
C****************************************************************************************************
c      WRITE(6,*)' IF YOU WANT TO CONTINUE ENETER ANY BUTTON'
c      READ(5,15)C
   15 FORMAT(A5)
   12 WRITE(6,13)
   13 FORMAT(20(/),13X,46('*'),/13X,'*',44(' '),'*',/,
     $13X,'*',4X,'  WILL YOU ENTER ANOTHER VALUES  ',7X,'*',/,
     $13X,'*',4X,'        IF YES  ENTER ---->  1  ',7X,'*',/,
     $13X,'*',4X,'        IF NOT  ENTER ----> ANY ',7X,'*',/,
     $13X,'*',44(' '),'*'/,13X,46('*'),10(/))
      READ(5,*) L
c      L=1
      IF(L.EQ.1) GO TO 5
      STOP
      END
C****************************************************************************************************
C* THIS SUBROUTINE SUBPROGRAM CALCULATES MONTHLY AVERAGE
C  SPHERICAL STORE TEMPERATURE BY ITERATION
C   METHOD AND ENERGY FRACTIONS
C****************************************************************************************************
```

```
      SUBROUTINE SIPSTORE(HT,RT,RN,HO,KTM,AA,BB,CC,Ac,R,Ta,RL,A,NOH,
     &FR,il,K,M,XCB,FBAR,QN,QSY,WY,QLT,QLY,QY,R1,R2,R4,R5,TALM,TNEW)
C****************************************************************************************************
      REAL FIW(12),FW(12),H(12),QN(12),TAG(12),KTM(12),R(12),RT(12),
     &RN(12),HO(12),FBAR(12),AA(12),BB(12),CC(12),QUM(12),XCB(12),
     &TALM(12),HT(12),TIN(12),RJ(13),NOH,S(12),DTT(12),ITC(12),TOLD(12),
     &TNEW(12),SBT(12),CP(5),CON(5),ALFA(5),WO(12),UL(4),RO(5),Ta(5,12),
     &Ef(12),Htt(12)
      Real UA(5),Td(5)
      INTEGER IMO(12)
      CHARACTER*12 AYLR(12)
      DATA SBT/0.,0.,0.,0.,1.,1.,1.,1.,1.,1.,0.,0./
      DATA RO/2050.,1500.,1500.,2640.,2500./,CP/1.842,.88,.8,.82,.90/,
     $CON/0.519,1.4,0.3,3.,1.3/,
     &ALFA/1.39E-7,1.1E-6,2.5E-7,1.4E-6,5.75E-7/
      DATA IMO/31,31,30,31,30,31,31,28,31,30,31,30/
      DATA AYLR/'JULY','AUGUST','SEPTEMBER','OCTOBER','NOVEMBER',
     $'DECEMBER','JANUARY','FEBRUARY','MARCH','APRIL','MAY','JUNE'/
      DATA Tia,Ti,Tinf,Pi,U/20.,15.,15.,3.1415,1.2/
      DATA Td/-12,-12,-9,-3,0/UL/7.4,4.5,5.,3.2/
C***************************************************************************************************
      UA(il)=10000/(Tia-Td(il))
      Fi=Tia/Tinf-1.
      P=1000.*4.187/(3.*RO(K)*CP(K))
      Y=ALFA(K)*31536000./(A*A)
      GAMA=4.*PI*A*CON(K)/(NOH*UA(il))
      HCONS=NOH*UA(il)/(4.*PI*A*CON(K)*Tinf)
      SCONS=Noh/(24.*3600.*4.*PI*A*CON(K)*Tinf)
C TIN;INITIAL STORE TEMPERATURE
C TAG;DIMENSIONLESS TIME
C H  ;DIMENSIONLESS HOUSE HEAT LOAD
C DTT;YEARLY DIMENSIONLESS TIME
C CALCULATION OF MONTHLY AVERAGE DIMENSIONLESS HOUSE HEAT LOAD
      DO 1 I=1,12
      H(I)=SBT(I)*HCONS*(Tia-Ta(il,I))
      IF(H(I).LE.0.0) H(I)=0.0
      TAG(I)=-1./2.+I/12.
      Dtt(i)=Alfa(K)*3600.*24.*IMO(I)/(A*A)
      TIN(I)=Ti
      TOLD(I)=Ti
    1 CONTINUE
      DO 2 J=1,13
      RJ(J)=(J-7)/12.0
    2 CONTINUE
C***************************************************************************************************
      Itr=1
C TOP OF ITERATION
    3 Itr=Itr+1
C***************************************************************************************************
C     CALCULATION OF MONTHLY AVERAGE HEAT PUMP WORK
C***************************************************************************************************
      DO 5 I=1,12
      Fiw(i)=Tin(i)/Tinf-1.
      Fa=Ta(il,I)/Tinf-1.
      C1=(Tinf*(Fiw(i)+1.)+100)/70
      Th=Tinf*(U*(Fi-Fa)+Fi+1)
      c2=Tinf*(U*(Fi-Fa)+Fi-Fiw(i))
      C3=(35.-Tinf*(U*(Fi-Fa)+Fi+1))/40.
      IF(C2.LE.0.0)Go to 4
      WO(i)=SBT(I)*(Fi-Fa)/(C1*ALOG((Th+273.)/C2)+C3)
      Go to 25
```

```
  4 WO(i)=0.0
 25 WO(I)=WO(I)/gama
  5 CONTINUE
C*****************************************************************************************************
C    CALCULATION OF THE MONTHLY AVERAGE ENERGY INPUT TO THE STORE
C*****************************************************************************************************
C S  ;DIMENSIONLESS SOLAR HEAT GAIN
C QN ;DIMENSIONLESS NET ENERGY INPUT TO THE STORE
      Q0=0.0
      DO 6 I=1,12
      ITC(I)=UL(M)*(Tin(i)-Ta(il,i))/TALM(I)
      IF(ITC(I).LE.0.0)ITC(I)=0.0
      XCB(I)=ITC(I)/(RT(I)*RN(I)*HO(I)*KTM(I))*0.0036
      FBAR(I)=EXP((AA(I)+BB(I)*RN(I)/R(I))*(XCB(I)+CC(I)*XCB(I)**2))
      Qum(I)=AC*FR*TALM(I)*HT(I)*FBAR(I)*10**6.
      Ef(i)=Qum(i)*100./(AC*HT(I)*10**6.)
      Htt(i)=Ht(i)*10**6./(24.*3600.)
      S(I)=Scons*Qum(I)
      QN(I)=S(I)-H(I)+WO(I)
      Q0=Q0+QN(I)
    6 CONTINUE
      A0=Q0/12.
C**************************************************************************************************
C    CALCULATION OF STORAGE TEMPERATURE BY COMPLEX FINITE FOURIER
C    TRANSFORM TECHNIC
C**************************************************************************************************
      DO 10 I=1,12
      SUMG=0.0
      DO 8 n=1,40
      ETA1=1.+SQRT(n*Pi/Y)
      ETA2=SQRT(n*Pi/Y)+P*2.*PI*n/Y
      ACNS=1.0/(Pi*n*(ETA1**2.+ETA2**2.))
      SUMA=0.0
      SUMB=0.0
      DO 7 J=1,12
      ETA3=SIN(2.*Pi*n*RJ(J+1))-SIN(2.*Pi*n*RJ(J))
      ETA4=COS(2.*Pi*n*RJ(J+1))-COS(2.*Pi*n*RJ(J))
      SUMA=SUMA+QN(J)*(ETA1*ETA3+ETA2*ETA4)
      SUMB=SUMB+QN(J)*(ETA2*ETA3-ETA1*ETA4)
    7 CONTINUE
      SUMG=SUMG+ACNS*(SUMA*COS(2.*Pi*n*TAG(I))+SUMB*SIN(2.*Pi*n*TAG(I)))
    8 CONTINUE
      FW(I)=A0+SUMG
      TNEW(I)=(FW(I)+1.)*Tinf
      WRITE(6,9) AYLR(I),TNEW(I)
    9 FORMAT('  Tnew(',A3,')=',F10.2)
   10 CONTINUE
      WRITE(6,11)Itr,A,Ac
   11 FORMAT('                    Itr=',I3,'   A=',F5.1,'  Ac=',F5.1)
C*****************************************************************************************************
C Check for Tnew and Told BY 0.01 Criterian
      DO 12 I=1,12
      FRK=ABS(TNEW(I)-TOLD(I))
      IF(FRK.GT.0.01) GO TO 13
   12 CONTINUE
      GO TO 15
C*****************************************************************************************************
C Calculation of new temperature using Relaxiation constant
   13 DO 14 I=1,12
      TIN(I)=TIN(I)+RL*(TNEW(I)-TIN(I))
      TOLD(I)=TNEW(I)
```

```
 14 CONTINUE
  GO TO 3
C END OF ITERATION
C*******************************************************************************************************************
 15 QY=0.0
  WY=0.0
  QSY=0.0
  QLY=0.0
  Tmn=0.0
  DO 16 I=1,12
  Tmn=Tmn+Tnew(i)
  QY=QY+QN(I)*DTT(I)
  QSY=QSY+S(I)*DTT(I)
  QLY=QLY+H(I)*DTT(I)
  WY=WY+WO(I)*DTT(I)
 16 CONTINUE
  QLT=QY
  F=(Qsy-Qlt)*100./Qly
  Tmn=Tmn/12.
  Write(1,*)'Monthly Average Tilted Radiation'
  Do i=1,12
  Write(1,28)Htt(i)
  Write(6,28)Htt(i)
 28 Format(F6.2)
  Enddo
  Write(1,*)'Monthly Average Collector Efficiency'
  Do i=1,12
  Write(1,28)Ef(i)
  Enddo
  Write(1,*)'Monthly Average Store Temperature'
  Do i=1,12
  Write(1,28)Tnew(i)
  Enddo
  Write(1,*)'Annually Average Store Temperature'
  Write(1,29)Tmn,Ac
  Write(6,29)Tmn,Ac
  Write(1,*)'Annually Average Solar Fraction'
  Write(1,29)F,Ac
  Write(6,29)F,Ac
 29 Format(F7.2,' Ac=',F5.0)
  RTOT=QSY+WY
  R1=QSY/RTOT
  R2=WY/RTOT
  R4=QLT/RTOT
  R5=QLY/RTOT
  RETURN
  END
```

VITA

Recep Yumrutaş was born in Pazarcık, Kahramanmaraş, in 1962. He received a B. Sc. and M. Sc. in Mechanical Engineering from the Middle East Technical University (METU), Gaziantep Extension Campus and University of Gaziantep, in 1988 and 1991, respectively.

Since 1989, he has been a research assistant at the Mechanical Engineering Department, University of Gaziantep.

His current research interests include solar energy, storage of solar energy, heat transfer, and heat pump. He is married and has a son.